**1，GitLab安装**

如果是root登录安装，则下面的所有命令可以去掉sudo关键字。

1.安装一些依赖软件包，SSH一般系统是默认安装好的，不过也不排除一些最小安装的系统没有sshd服务。

sudo yum install -y curl policycoreutils-python openssh-server （-y参数静默安装，系统不会询问）

sudo systemctl enable sshd

sudo systemctl start sshd

2.关闭防火墙，或者开放HTTP的端口并刷新防火墙

3.安装邮件服务，当gitlab想要通过邮件通知，也可以另外配置其它的邮件服务器

sudo yum install postfix

sudo systemctl enable postfix

sudo systemctl start postfix

4.从官网获取安装源的脚本，并直接执行脚本，如此，则yum命令可以找到gitlab-ee下载地址。

curl -s "https://packages.gitlab.com/install/repositories/gitlab/gitlab-ee/script.rpm.sh" | sudo bash

详细查阅：<https://docs.gitlab.com/omnibus/update/README.html>

可以将脚本文件先存储，自己执行脚本文件安装源地址：

curl -s "https://packages.gitlab.com/install/repositories/gitlab/gitlab-ee/script.rpm.sh" > /home/rpm.sh

chmod +x /home/rpm.sh

cd /home

./rpm.sh

5.安装gitlab

yum install -y gitlab-ee

可以查看gitlab安装在哪里

rpm -ql gitlab-ee | less

6.可以配置一下gitlab（此项也可以不做）

修改gitlab的配置文件：/etc/gitlab/ gitlab.rb

可以改变访问地址路径：

external\_url 'http://gitlab.repository.com‘

可以使用https：

external\_url 'https://gitlab.repository.com‘

可以更改端口（默认https是443，http是80）：

external\_url 'http://10.5.13.100:9000‘

注意如果服务器没有域名地址，请使用IP地址代替

改变监听端口还可以配置（如果同时配置，该项覆盖上面的external\_url配置）

nginx['listen\_port'] = 9000

如果上述配置了https

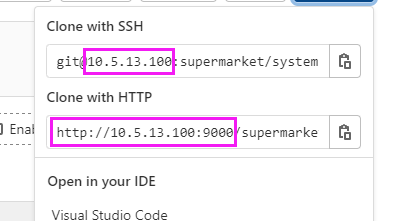
可以使用nginx['listen\_https'] = false关闭https，注意这里不能设置true来覆盖external\_url，否则gitlab的nginx服务器起不来

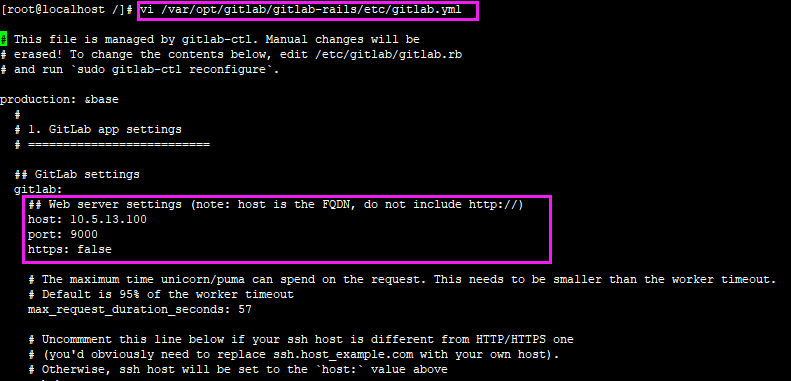
运行配置

sudo gitlab-ctl reconfigure

改完配置后运行此项可以自动改写/var/opt/gitlab/gitlab-rails/etc/gitlab.yml文件下的配置，但是注意这个只针对external\_url修改可以改写，修改nginx配置即使运行reconfigure还是会按照external\_url内容修改，所以要手动修改gitlab.yml文件，修改这个文件主要是改变代码库git地址。

详细查阅：<https://docs.gitlab.com/omnibus/settings/nginx.html>





重启gitlab

gitlab-ctl restart

可以查看gitlab-ctl命令的帮助信息

gitlab-ctl –help

7.打开浏览器输入地址查看效果,第一次打开页面会让我们设置root用户的密码。记住自己设置的密码，再次刷新进入登录页面，使用root账号登录进去。

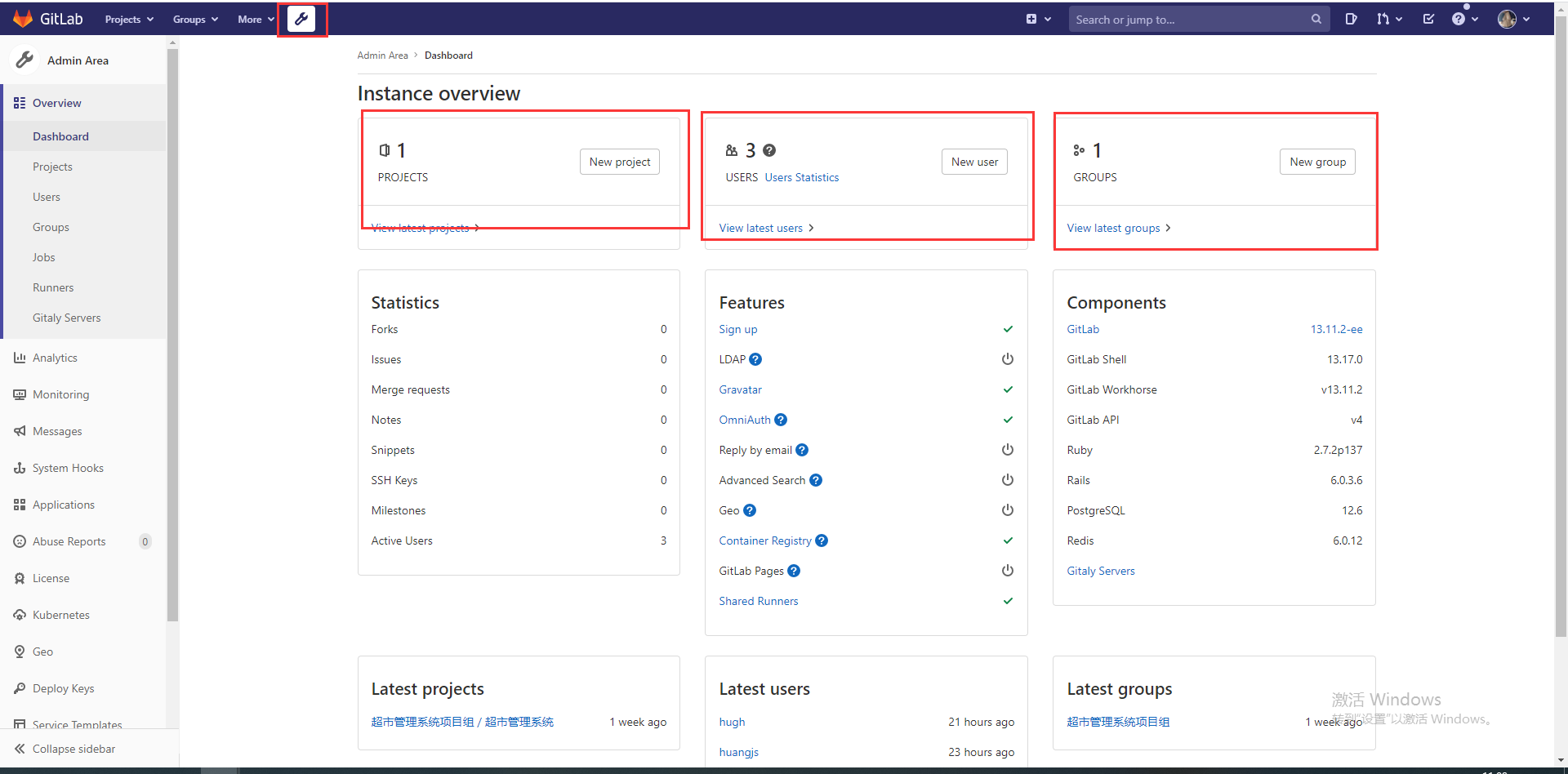
注意：如果访问不了，则参看是否操作了第二步骤，并且使用https试一试。如果浏览器访问gitlab的web页面非常慢，出现502问题，则下面两种情况试试：

1，使用netstat -ntpl查看端口情况（如果linux没有该命令，安装对应的netstart包）查看gitlab所有使用的端口是否被占用，如果占用，则可以kill被占用的端口进程，刷新页面，或者修改配置文件/etc/gitlab/gitlab.rb，更改端口（如何修改端口，自行查阅）。然后gitlab-ctl reconfigure gitlab-ctl restart

2，使用cat /proc/swaps查看swap分区存储是否够用（如何增加swap分区大小，自行查阅）

**2，GitLab基本配置使用**

GitLab安装后，初始访问会让设置root密码，设置完跳到登录页面，使用root登录，root是超级管理员，可以配置所有东西。先看主页：



点击左上角设置按钮，普通用户登录进去后是没有这个按钮的。

**Gitlab包含Group，Project，User(Member)三种概念**

Group是一个父子结构的目录（具有权限），就是一颗树，比如某个项目组，其下还有子模块组等。

Group每一级都可以设置关联多个User被称作Member，同时每一级都可以添加(Transfer)多个Project。

Group所关联的Member和Member对应的权限，均会继承到Group下的所有Project和子Group上。

Project就是一个个独立的项目（具有权限），比如超市系统项目。

Project可以被添加(Transfer)到某个Group下属于某个项目组的项目，同时也可以直接被添加(Transfer)到某个User下属于某个人的项目，同时只能被添加到一个里面。

Project中除了继承Group的Member，还可以单独设置关联多个User也称作Member。

User就是管理员创建的用户（普通用户(Regular)和管理员用户(Admin)），也可以是管理员自己。

User可以拥有自己的Project，比如个人项目，同时可以作为Member（具有权限）被添加到Group和Project中去，可以被添加到多个里面。

**Gitlab的Memebr在Group和Project中有五种权限：Guest、Reporter、Developer、Master、Owner**

Guest：可以创建issue、发表评论，不能读写版本库

Reporter：可以克隆代码，不能提交，QA、PM可以赋予这个权限

Developer：可以克隆代码、开发、提交、push，RD可以赋予这个权限

Maintainer：可以创建项目、添加tag、保护分支、添加项目成员、编辑项目，核心RD负责人可以赋予这个权限

Owner：可以设置项目访问权限 - Visibility Level、删除项目、迁移项目、管理组成员，开发组leader可以赋予这个权限

**Gitlab中的Group和Project有三种访问权限：Private、Internal、Public**

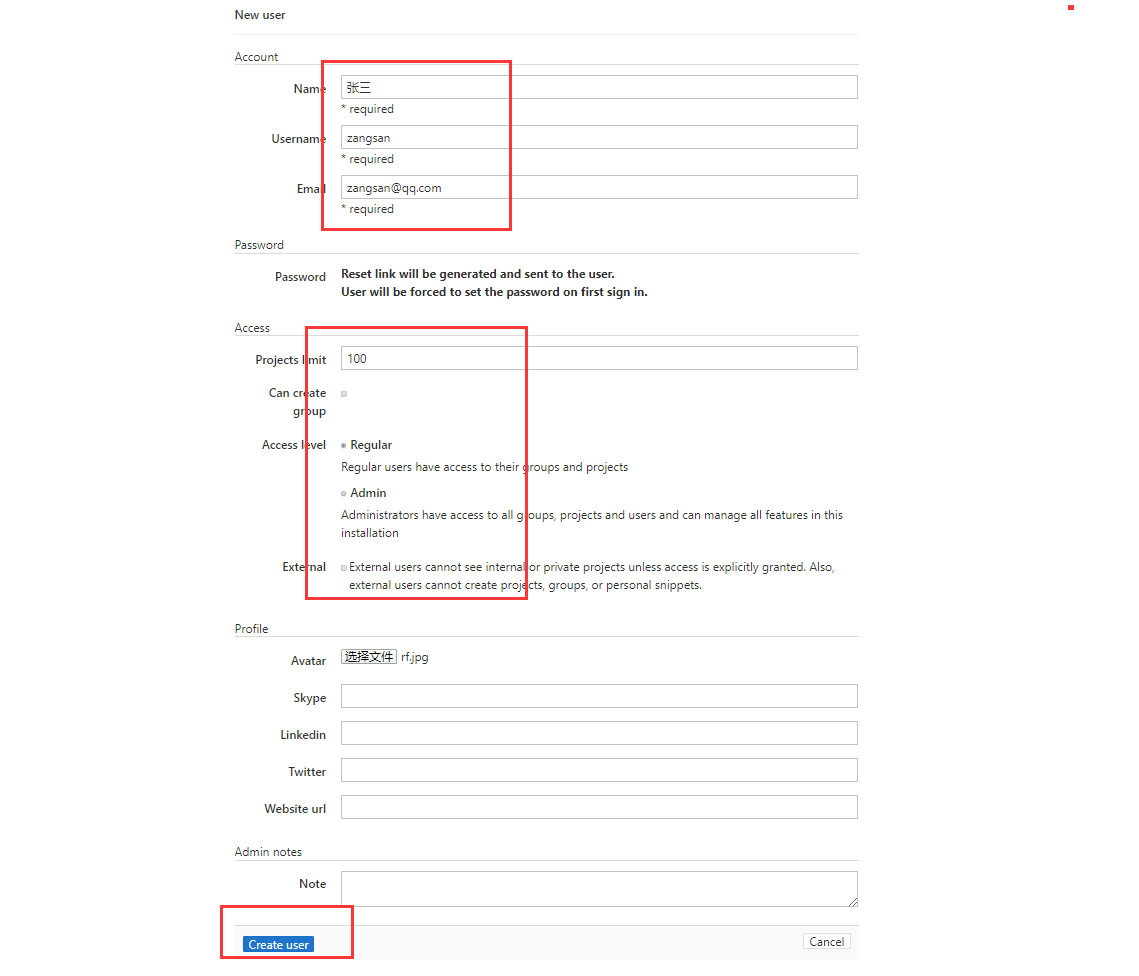
Private：只有组成员才能看到

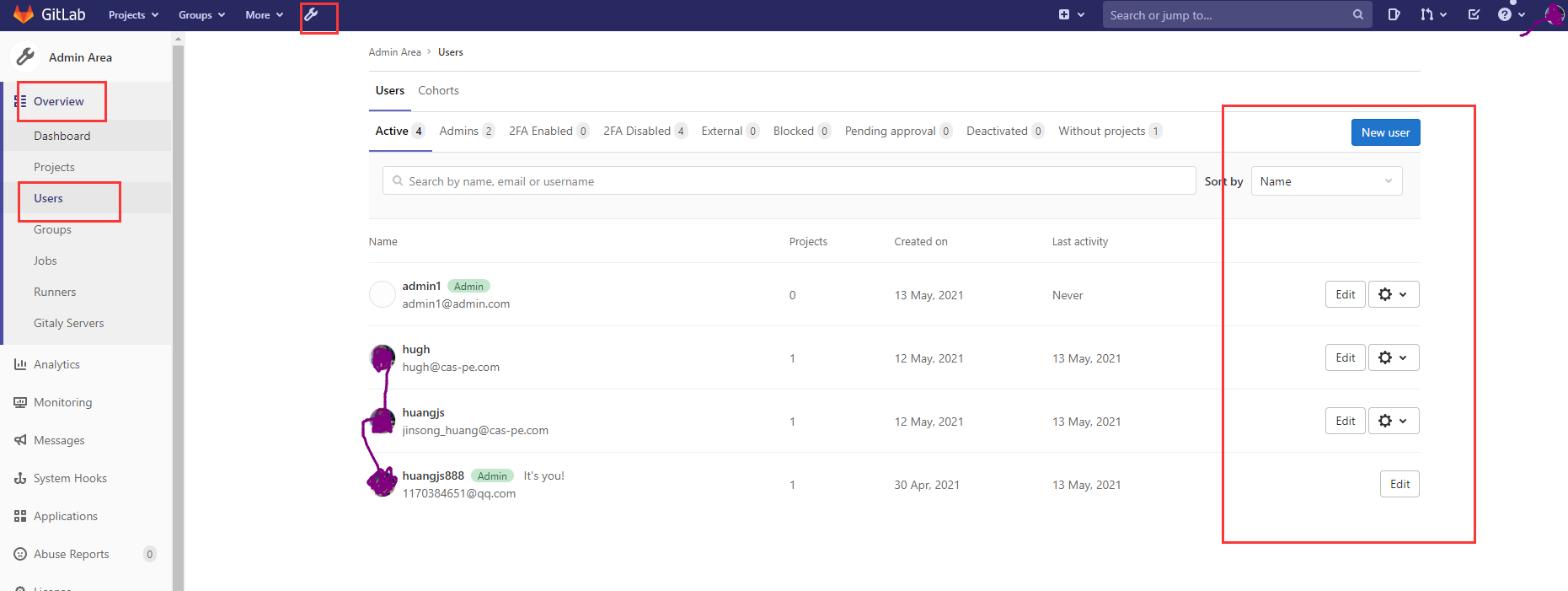
Internal：只要登录的用户就能看到，开源项目和组一半设置的是这个

Public：所有人都能看到

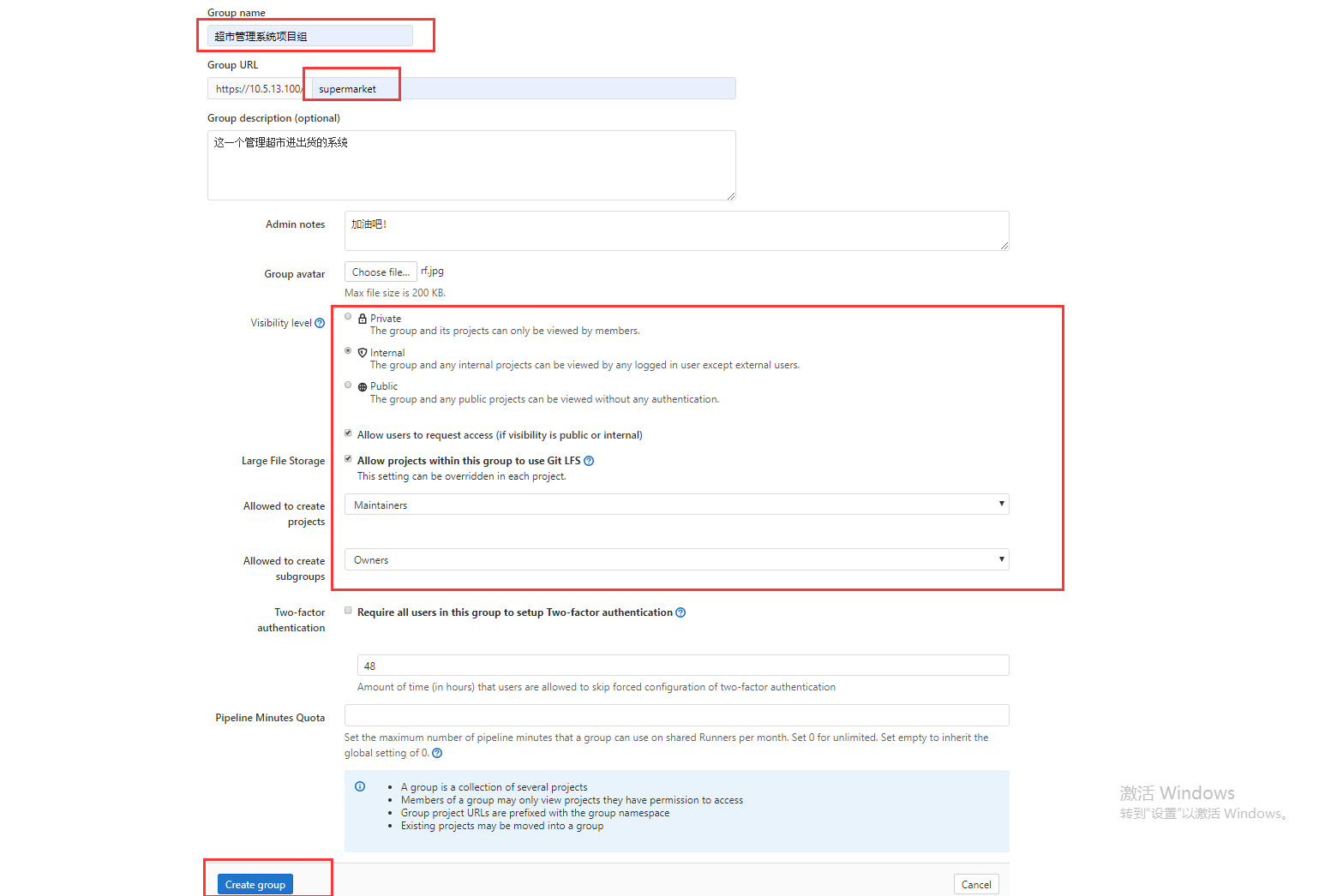
具体权限详情可以查阅gitlab相关文档，下面便是创建Group，Project，User，以及添加(Transfer)Member并设置相关权限了（GitLab安装好首次登录后会默对上述3个分别创建一个）

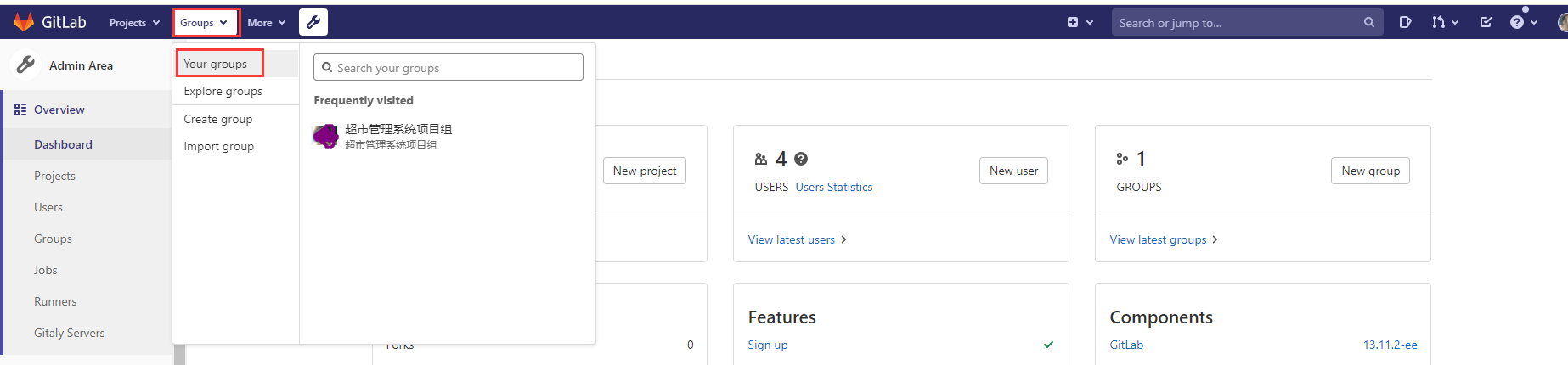
**1，主页设置里，点击New User按钮**

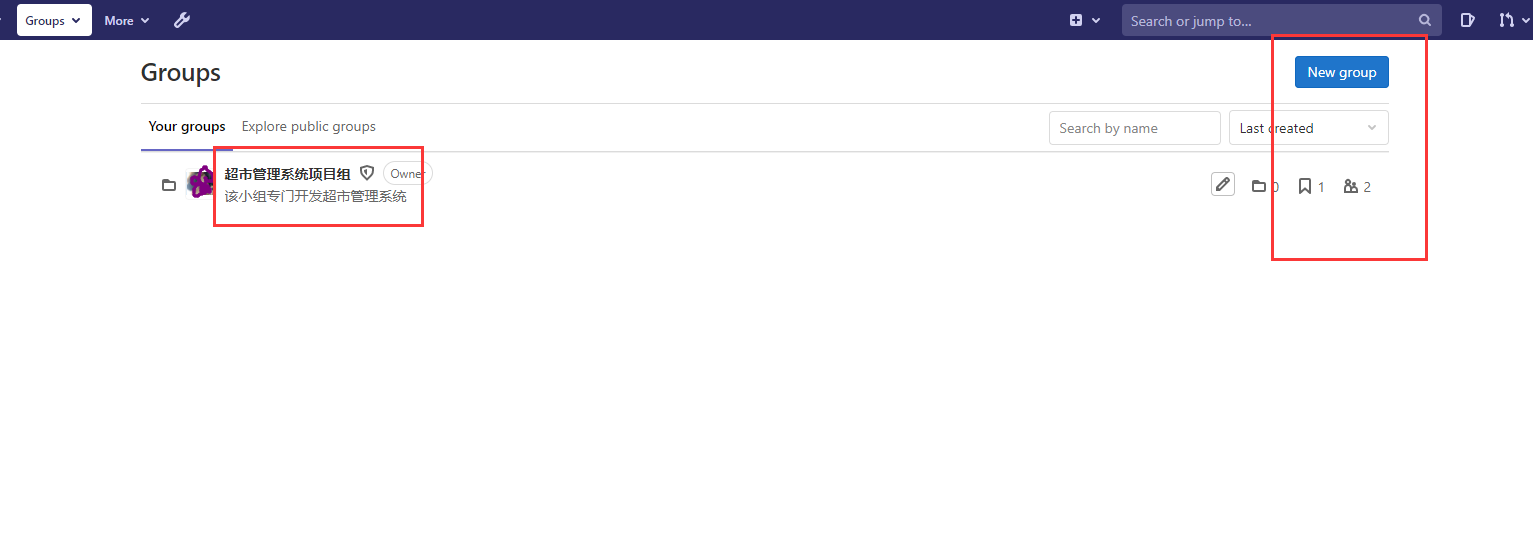




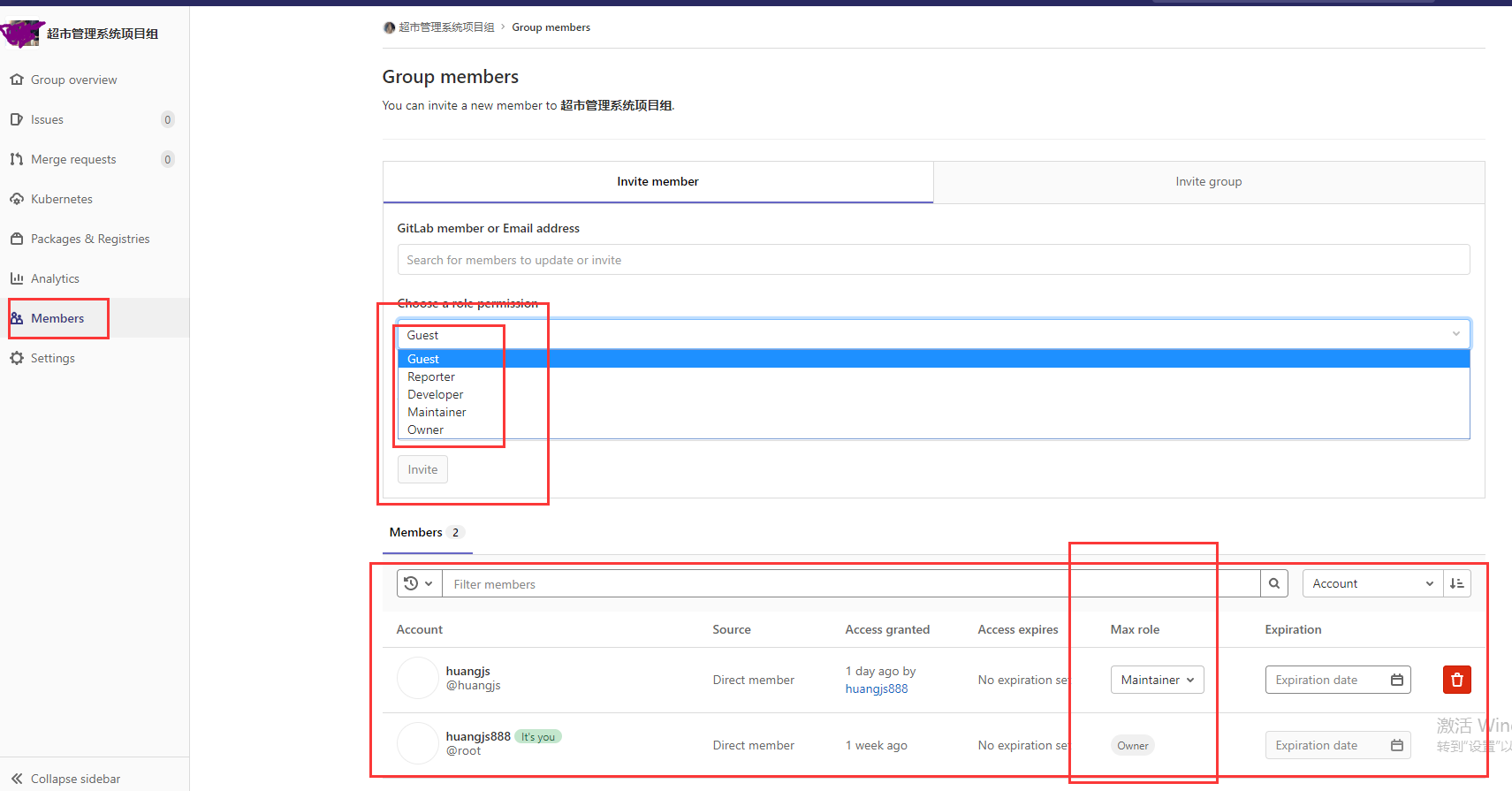
**2，主页设置里，点击New group按钮**



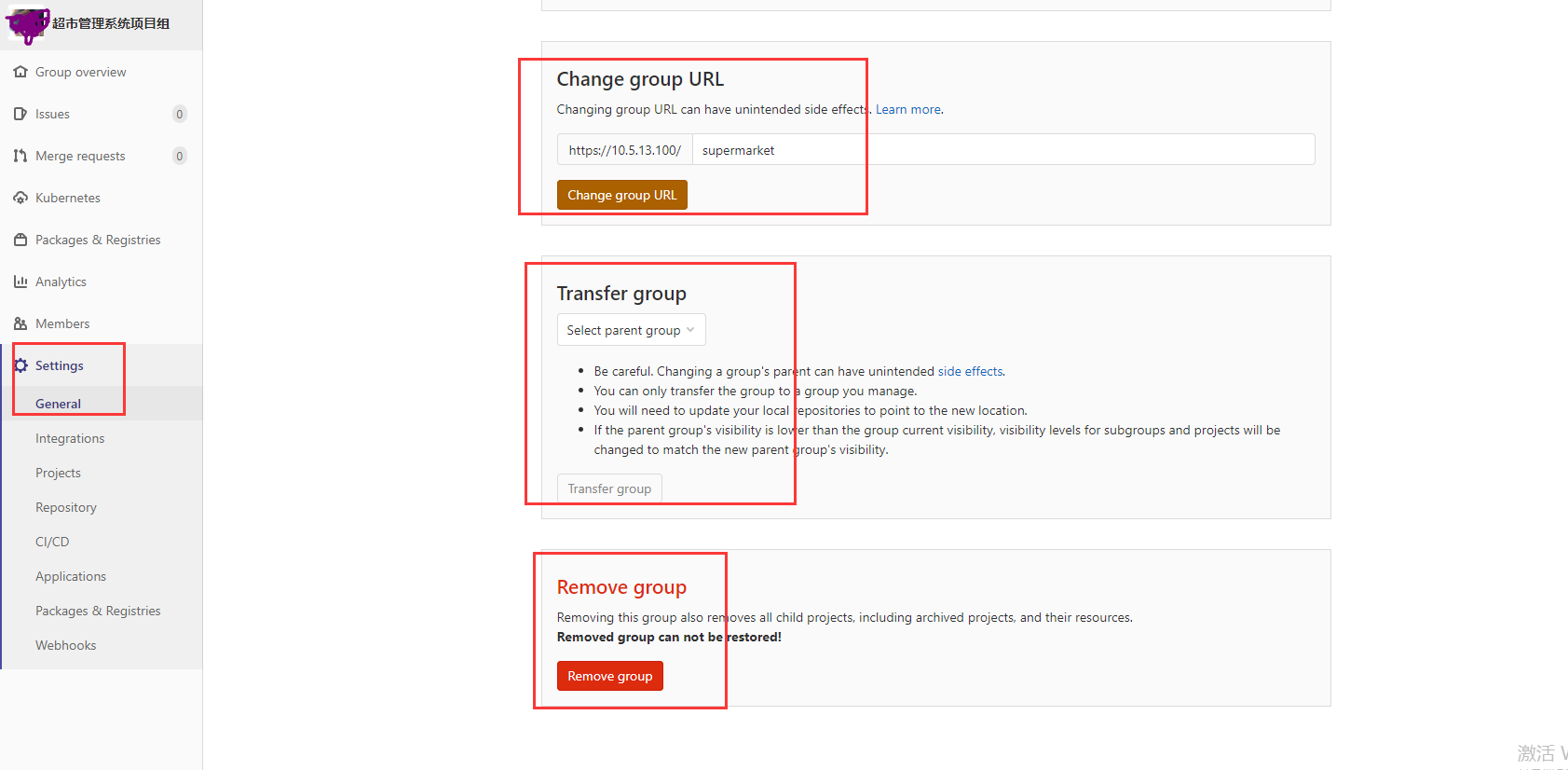




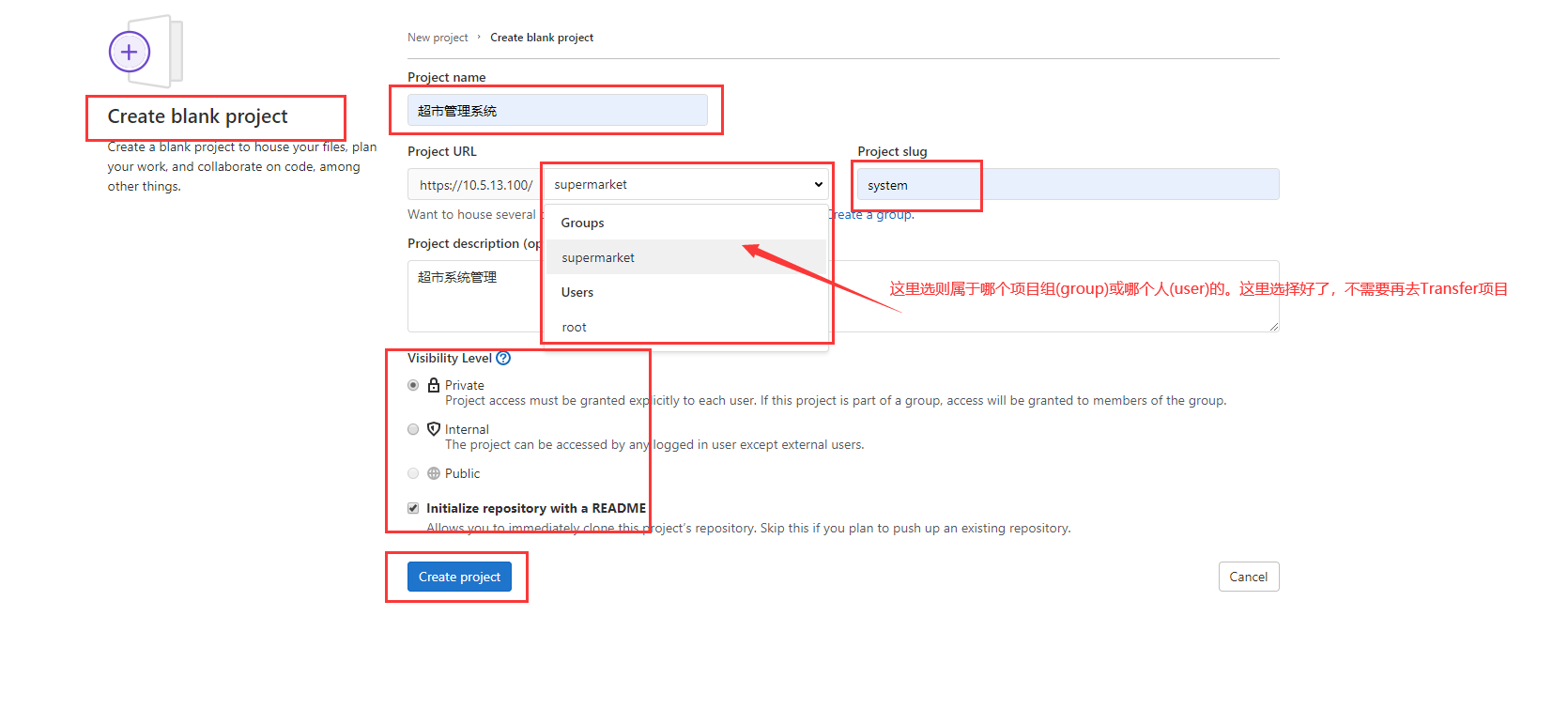
点击组名称进入该组，点击左侧Member，这里可以对项目添加User作为Member并赋予权限

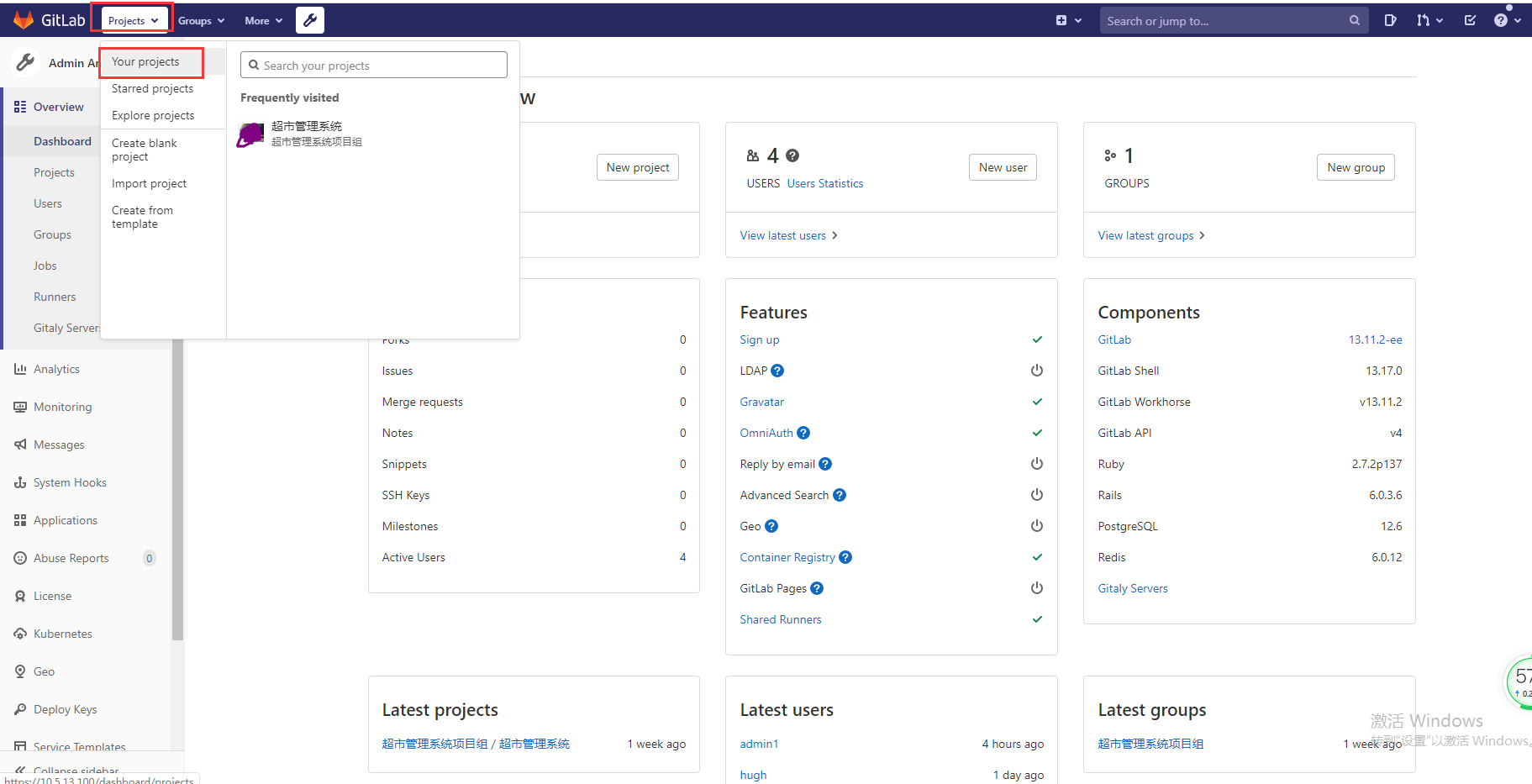


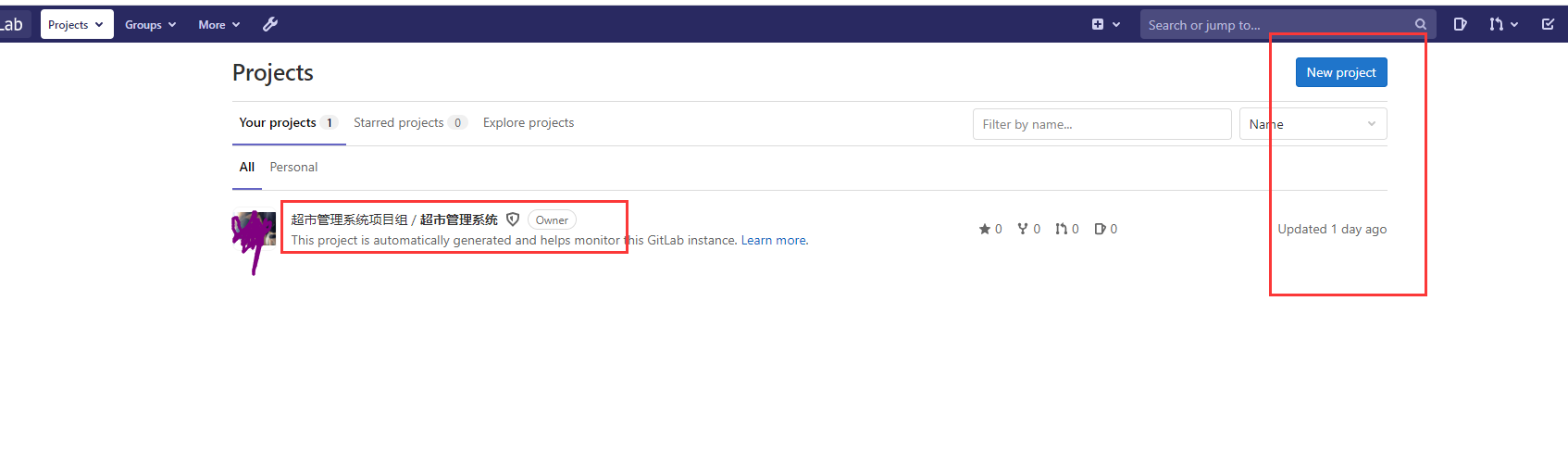
进入组后点击左侧Setting-General，下拉到最下面找到Advanced，点击右边Expand，然后可以改变组路径，删除组，以及将该组Transfer到另一个组下，成为子组。



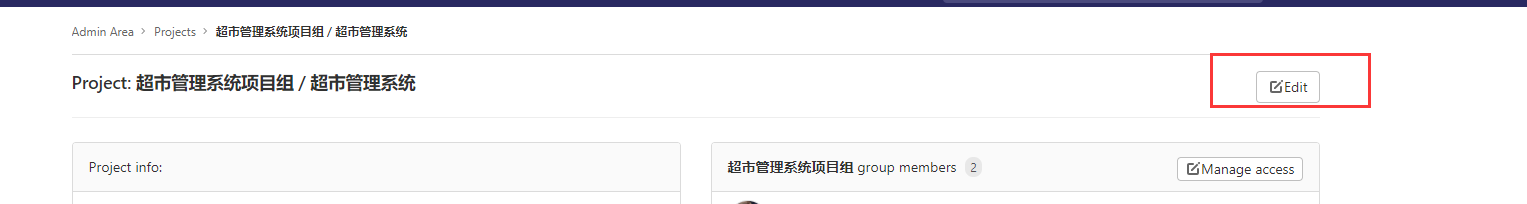
**2，主页设置里，点击New project按钮（会让选择创建什么方式（blank project，from template，import）的项目，我们选择blank project）**

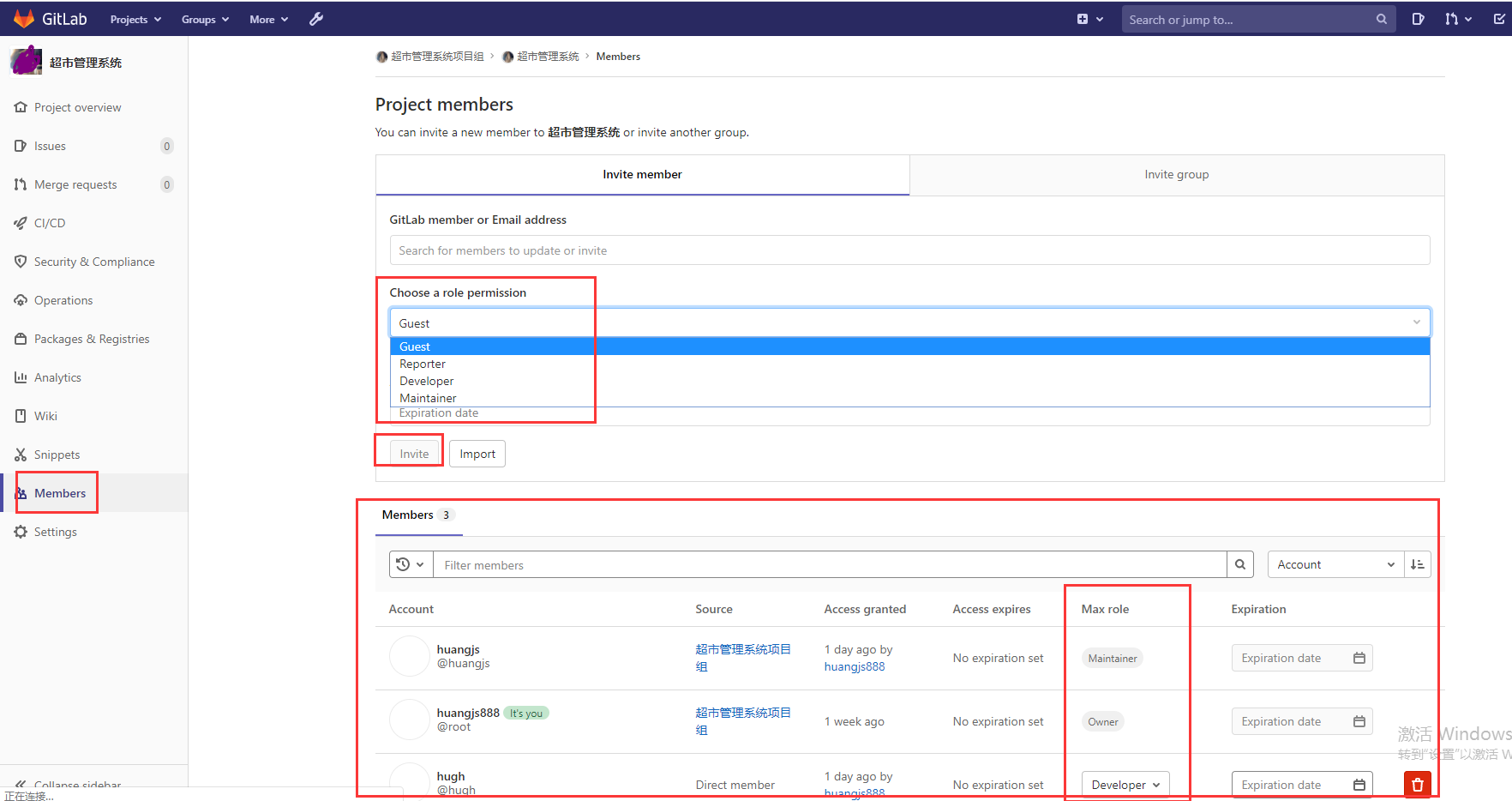




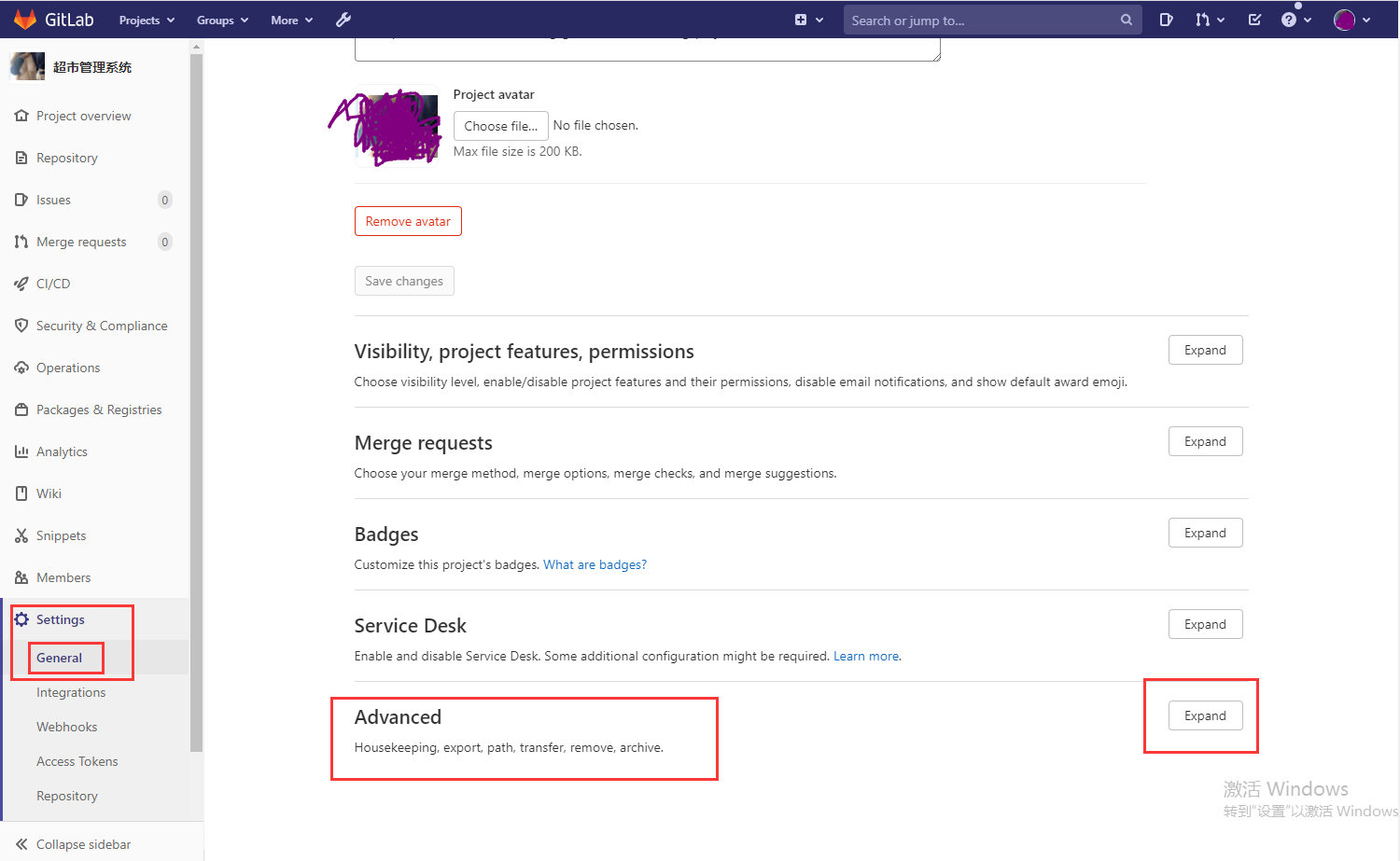


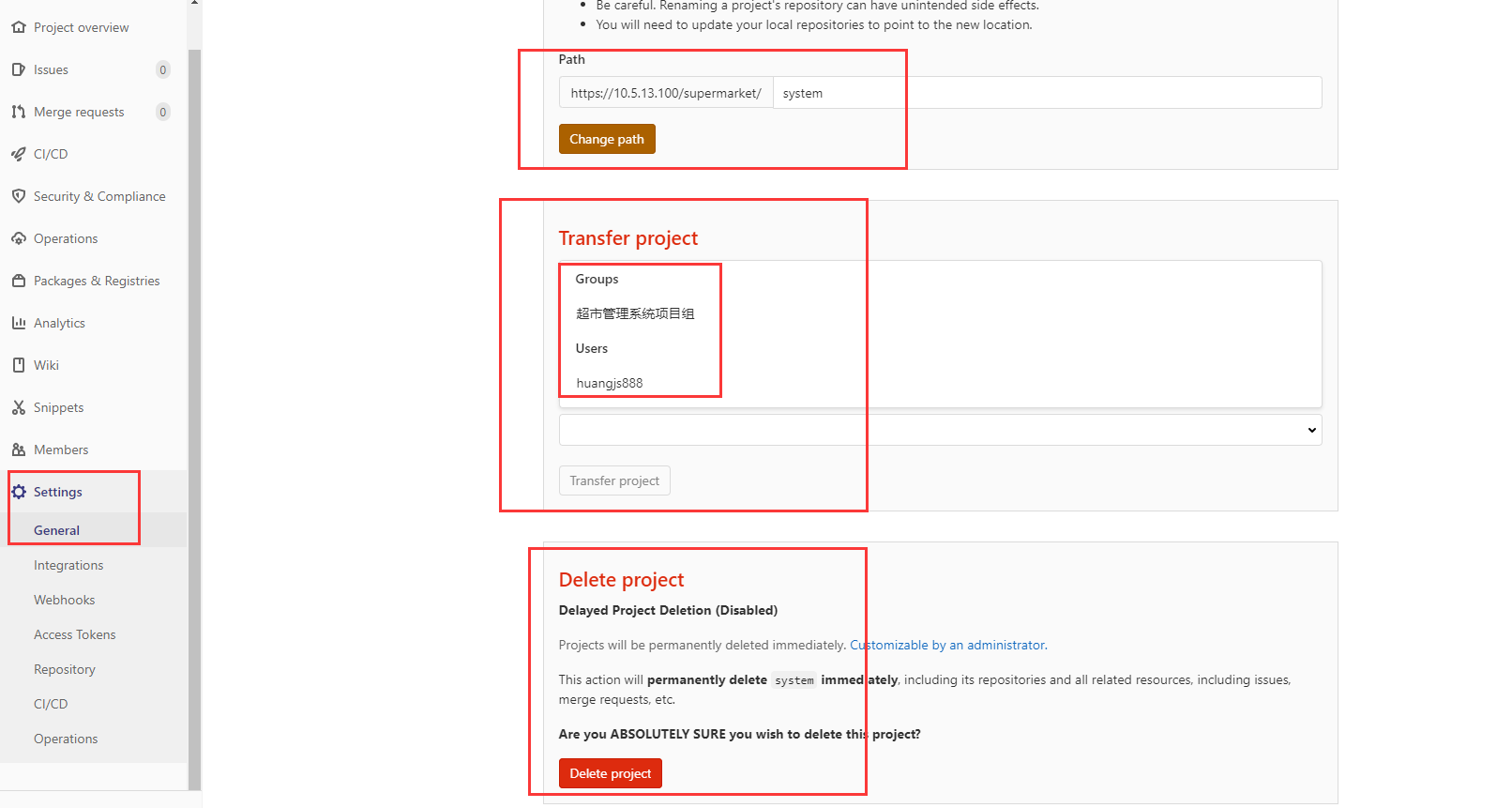
点击项目名称进入该项目，进入后点击左侧Member，这里可以对项目添加User作为Member并赋予权限



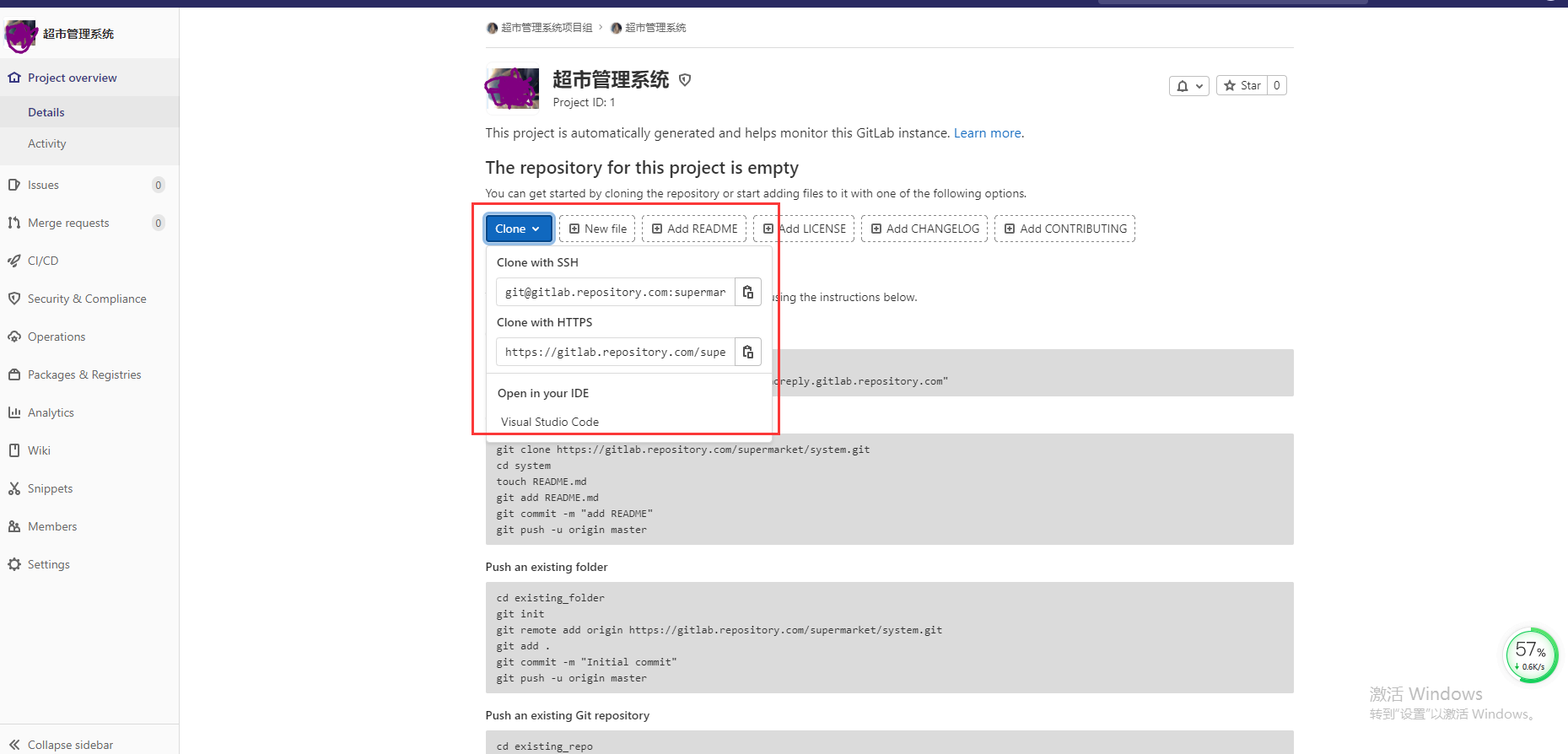


进入项目后点击左侧Setting-General，下拉到最下面找到Advanced，点击右边Expand，然后可以改变项目路径，删除项目，以及将该项目Transfer到Group或User下。

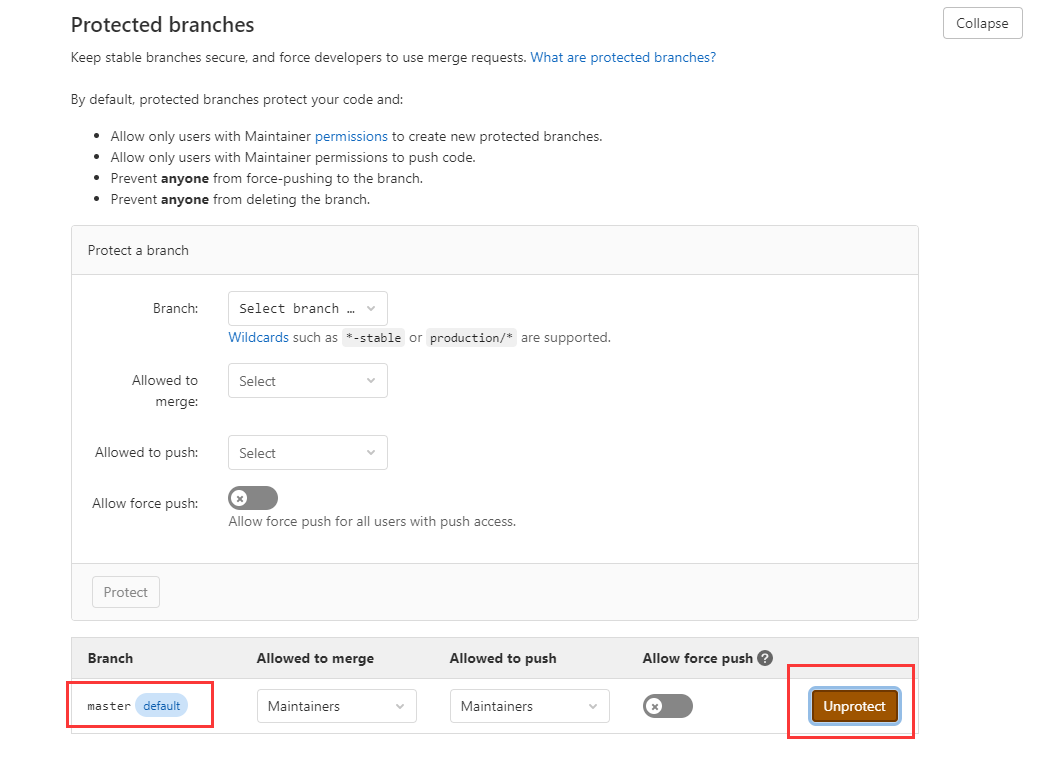
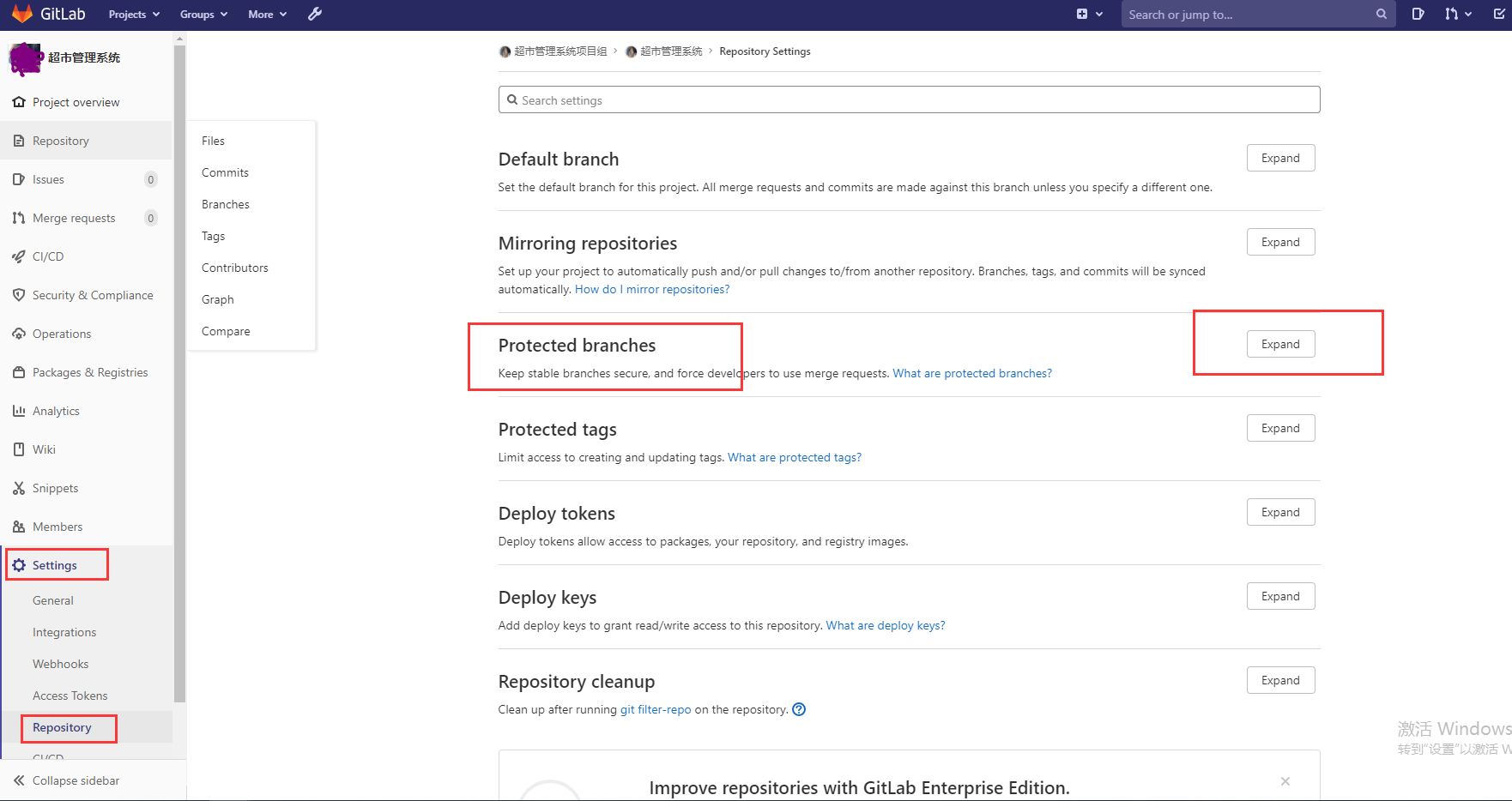




进入项目后，这里就可以使用该项目的https或ssh地址并使用git管理代码了，如何管理代码，请参考GitHub上传更新及删除项目文档，GitLab和GitHub使用大致相同，另外设置ssh的方式与GitHub也一样。



另外，对于Group和Project中设置为Developers的Member，其默认是没有权限push到master分支的，这里设置一下可以放开限制。进入项目后点击左侧Setting-Repositioy，找到Protected branches，点击右边Expand，然后点击master分支的Unprotect，或者将Allowed to merge和Allowed to push改为Developers+Maintainers即可。



**Git，GitHub与GitLab的区别**

Git是一种版本控制系统，是一种工具，用于代码的存储和版本控制。Git(读音为/gɪt/。)是一个开源的分布式版本控制系统，可以有效、高速地处理从很小到非常大的项目版本管理。

GitHub是一个基于Git实现的在线代码仓库，是目前全球最大的代码托管平台，可以帮助程序员之间互相交流和学习。是一个面向开源及私有软件项目的托管平台，因为只支持git作为唯一的版本库格式进行托管，故名GitHub。

GitLab是一个用于仓库管理系统的开源项目，使用Git作为代码管理工具，并在此基础上搭建起来的web服务。是一个基于Git实现的在线代码仓库软件，你可以用GitLab自己搭建一个类似于GitHub一样的仓库，但是GitLab有完善的管理界面和权限控制，一般用于在企业、学校等内部网络搭建Git私服。

GitHub和GiLlab两个都是基于Web的Git远程仓库，它们都提供了分享开源项目的平台，为开发团队提供了存储、分享、发布和合作开发项目的中心化云存储的场所。从代码的私有性上来看，GitLab 是一个更好的选择。但是对于开源项目而言，GitHub 依然是代码托管的首选。