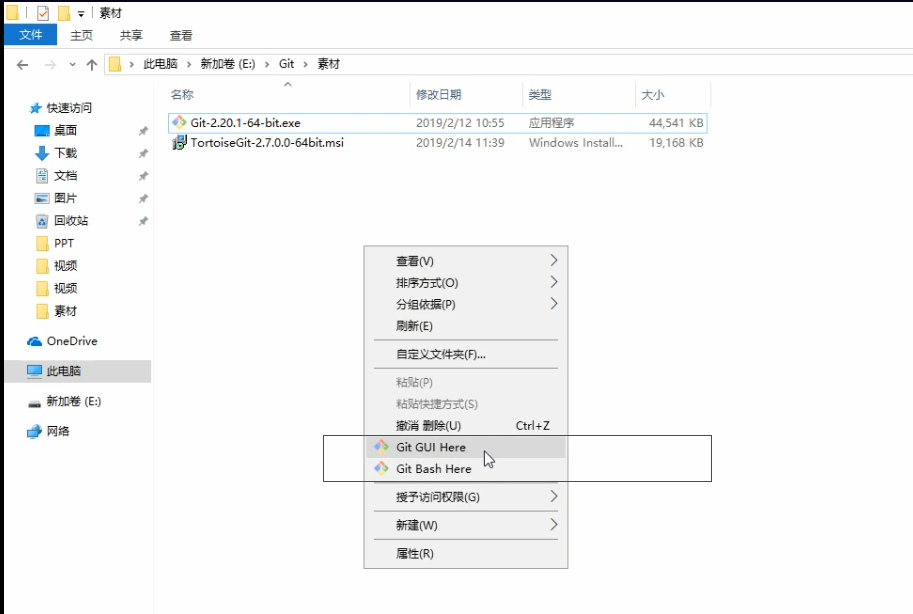
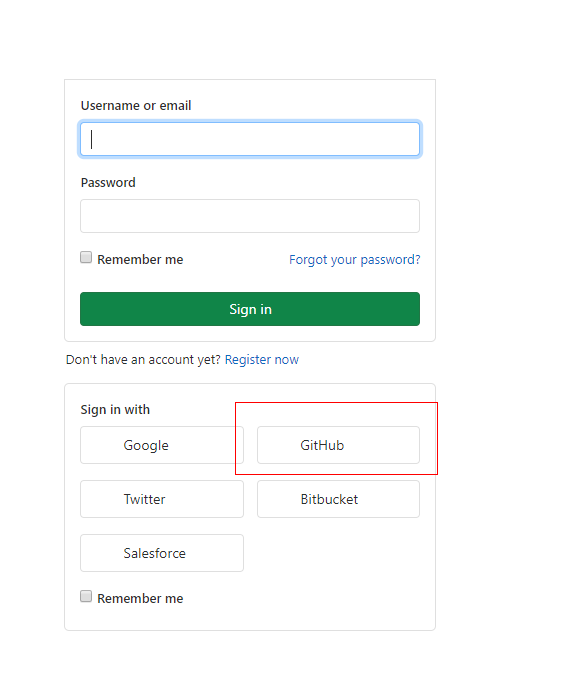
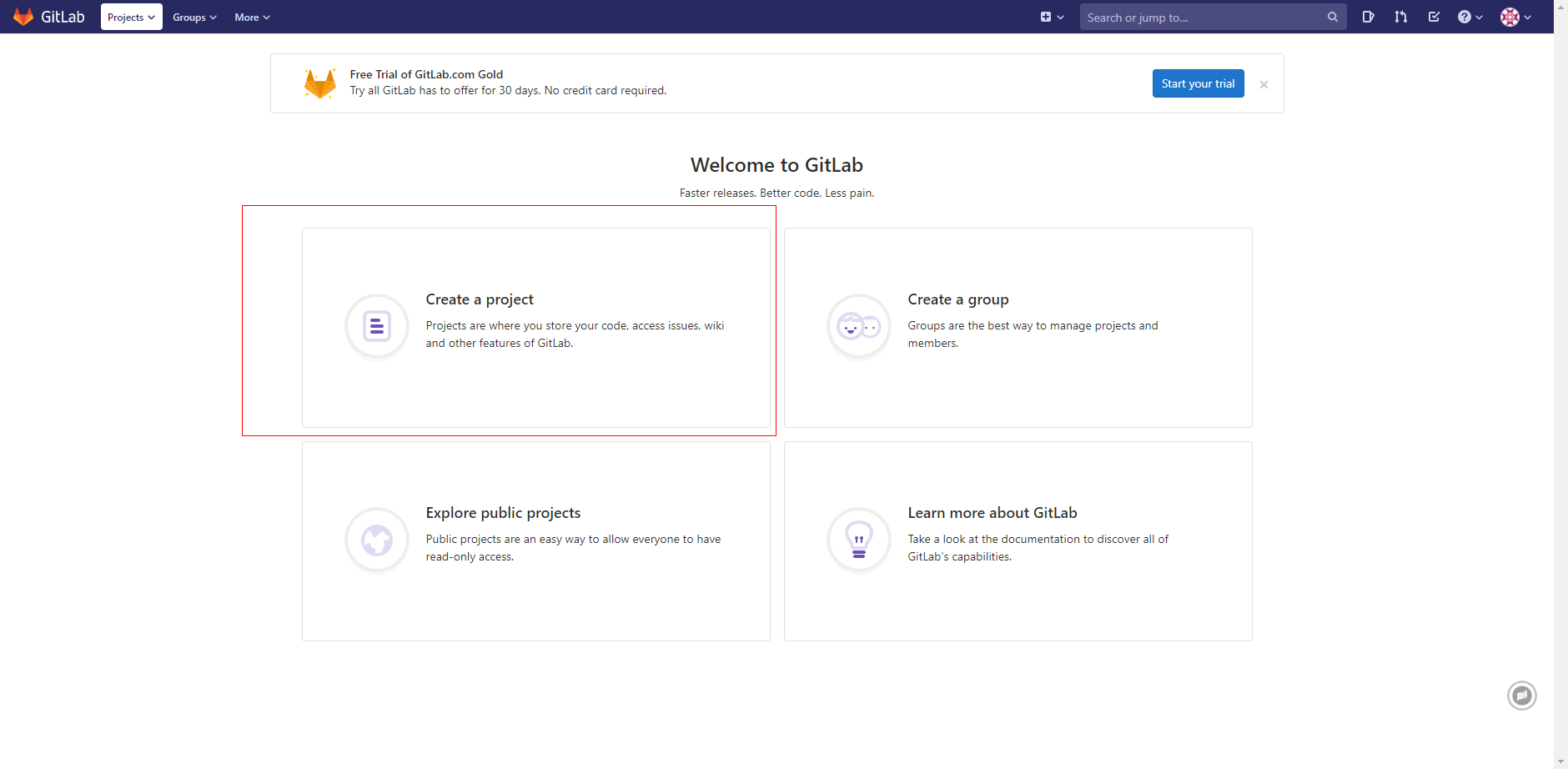


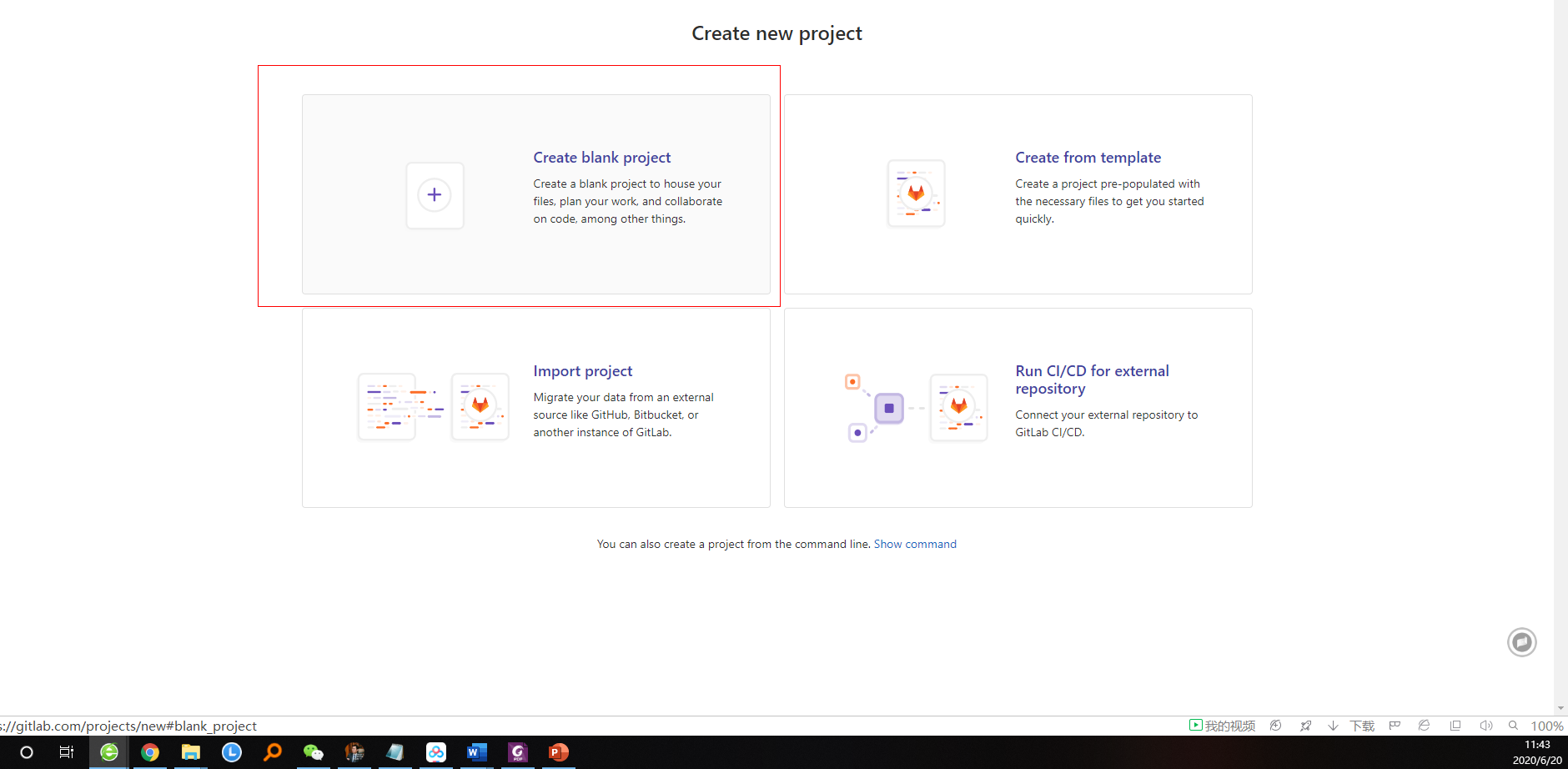
Git下载地址： <https://git-scm.com/downloads>

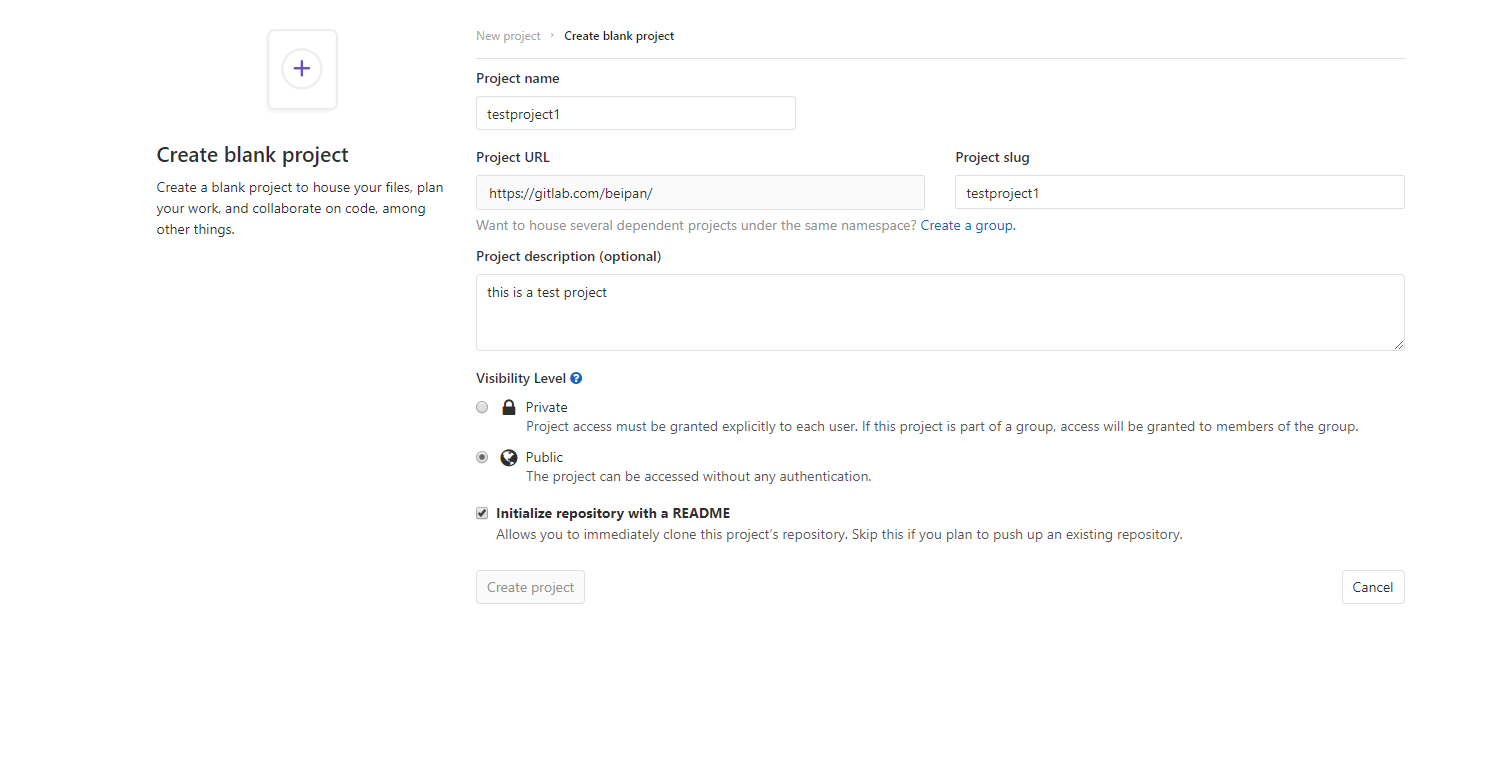
安装  
全部默认安装 一直点击下一步

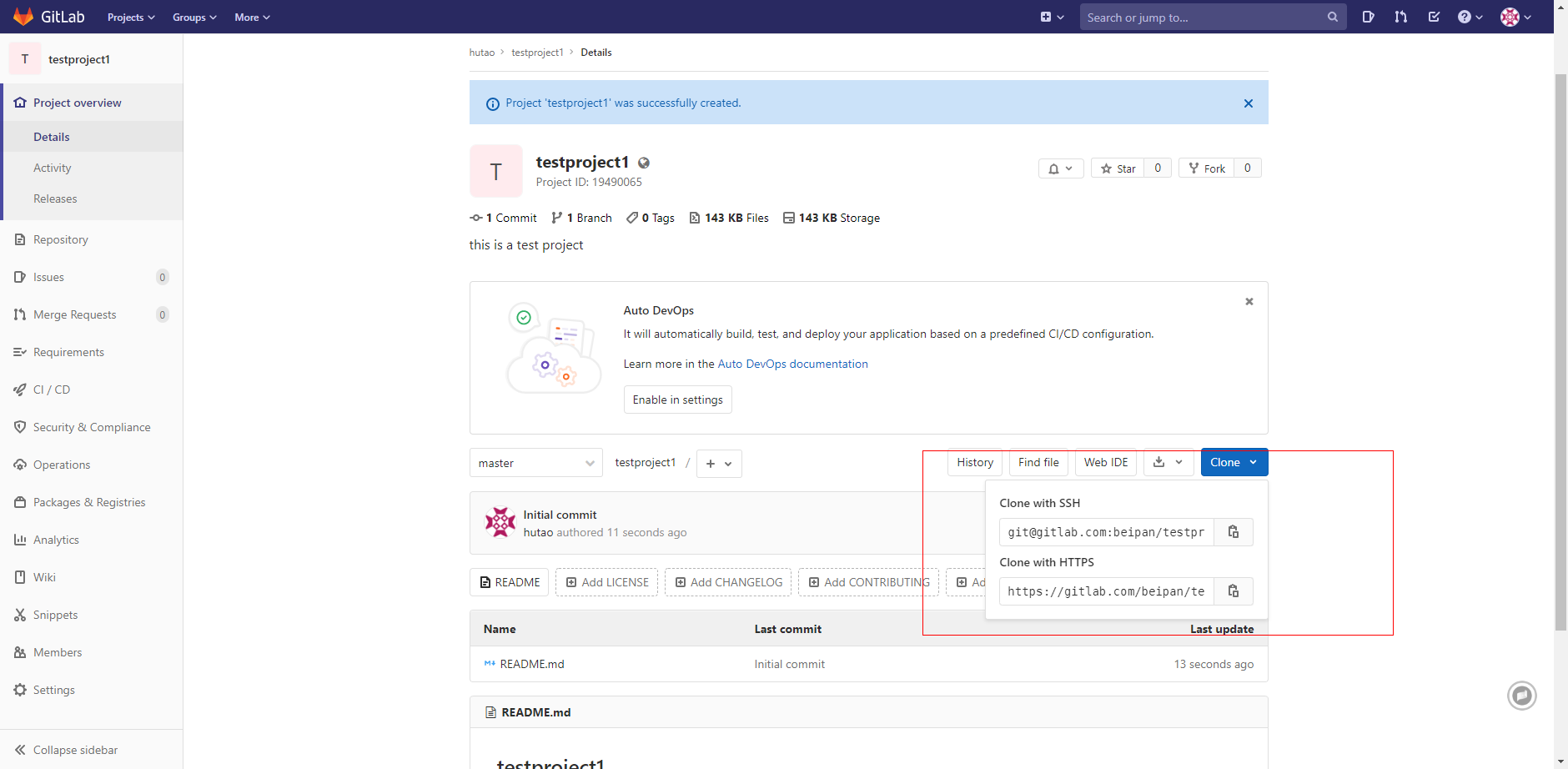
说明安装成功

gitlab注册登录  
用已有的github账号登录  
  
  
  
  
  
  
  
  
  
  
创建项目

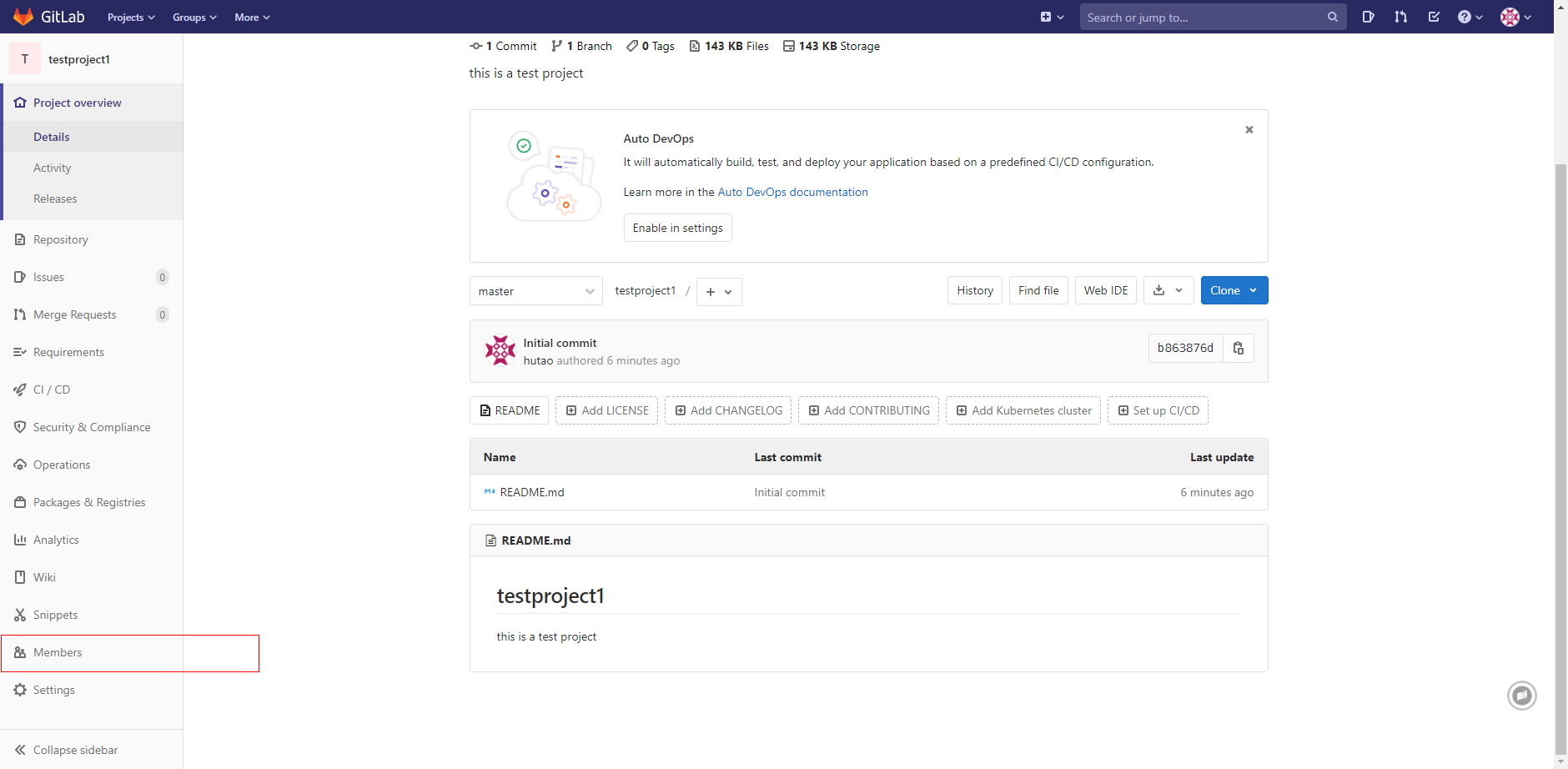




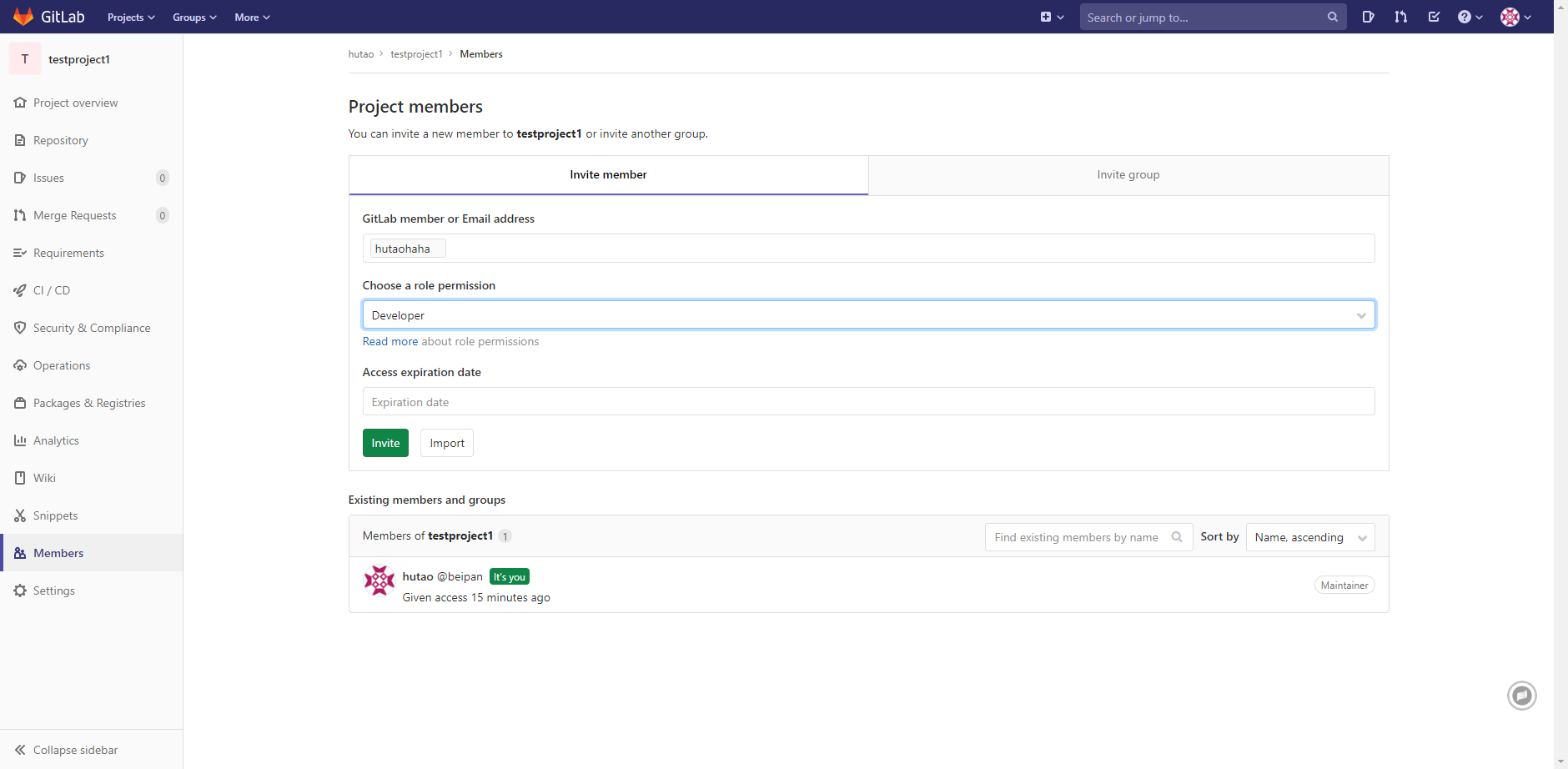




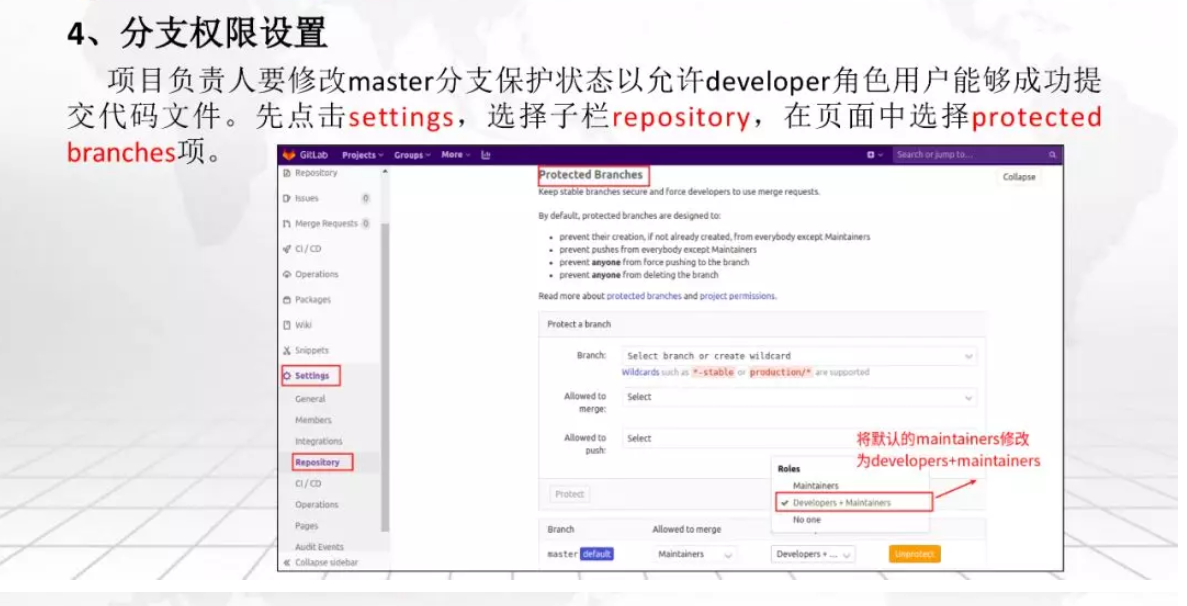
添加项目成员



添加项目成员



分支权限设置



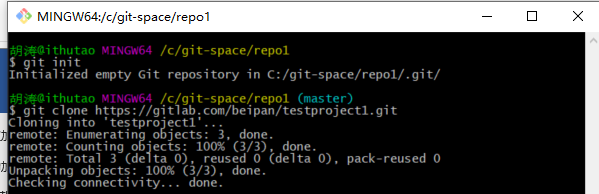
当安装Git后首先要做的事情是设置用户名称和email地址。这是非常重要的，因为每次Git提交都会使用该用户信息  
设置用户信息

git config --global user.name “hutao”

git config --global user.email “[1152253154@qq.com](mailto:1152253154@qq.com)”

查看配置信息  
git config –list

创建空目录 右键打开git clone https://gitlab.com/beipan/testproject1.git



git status 查看文件状态

在目录中新建一个hello.txt

git add 将未跟踪的文件加入暂存区:

git add hello.txt

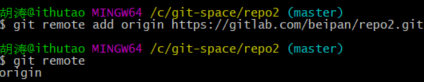
git commit 将暂存区的文件修改提交到本地仓库:

git commit -m “commit hello.txt”

git log 查看日志记录

如果想查看已经配置的远程仓库服务器，可以运行 git remote 命令

运行 git remote add 添加一个新的远程 Git 仓库，同时指定一个可以引用的简写



如果因为一些原因想要移除一个远程仓库 ，可以使用 :

git remote rm remote

注意：此命令只是从本地移除远程仓库的记录，并不会真正影响到远程仓库

--allow-unrelated-histories

git pull 是从远程仓库获取最新版本并merge到本地仓库:

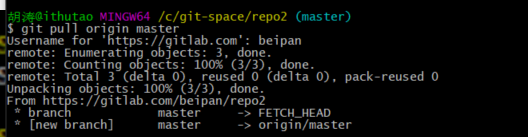
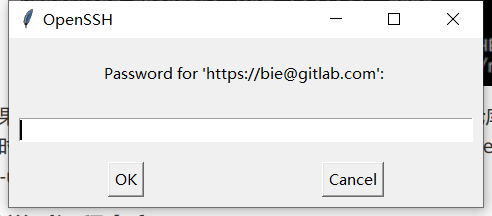
git pull origin master

(

Username:beipan

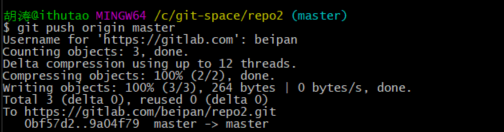
Password:ht1920190965

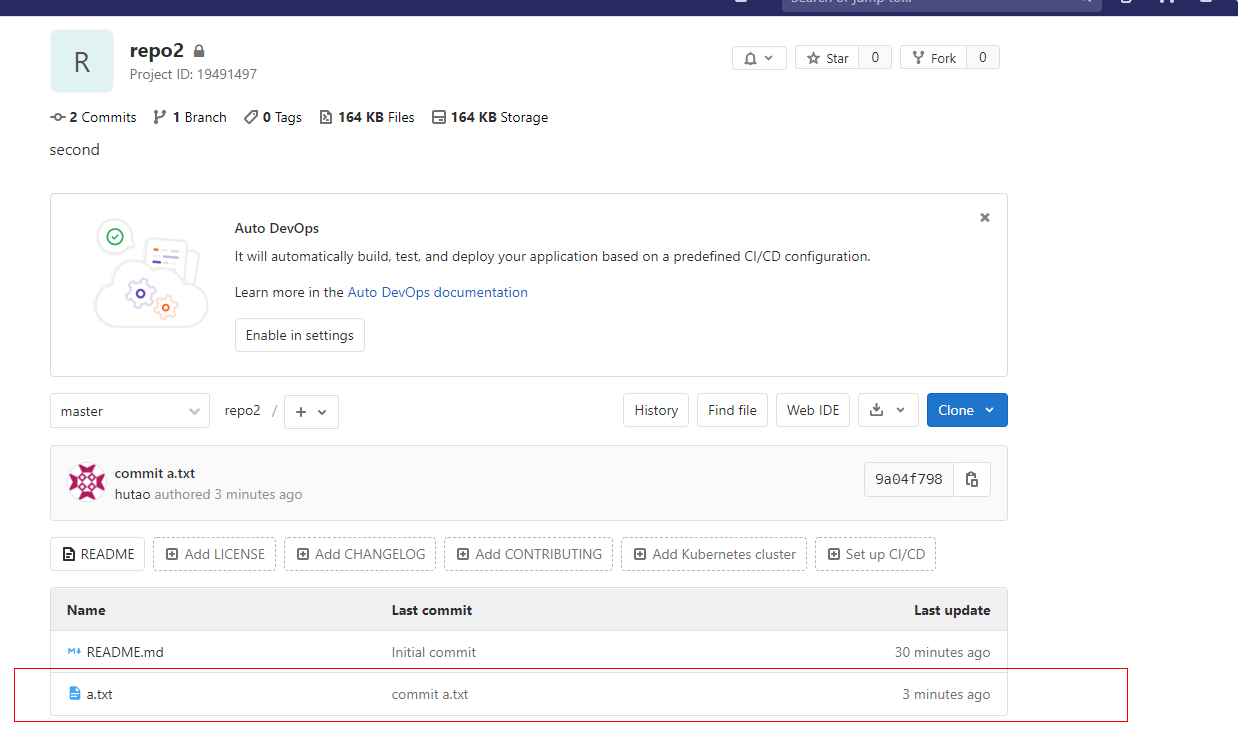
)

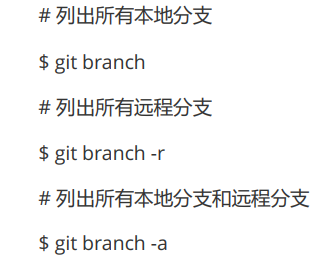
 

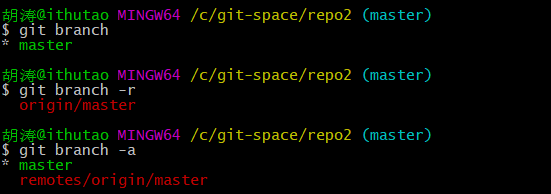
当你想分享你的代码时，可以将其推送到远程仓库

git push origin master

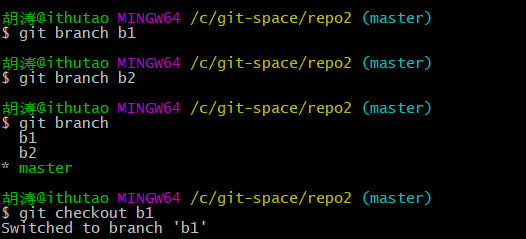






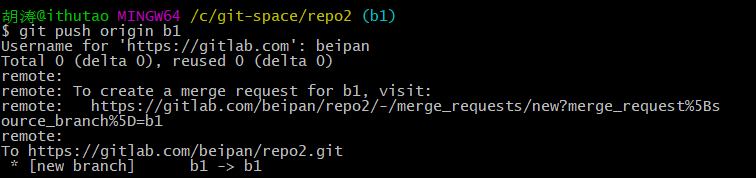


分支创建 分支切换

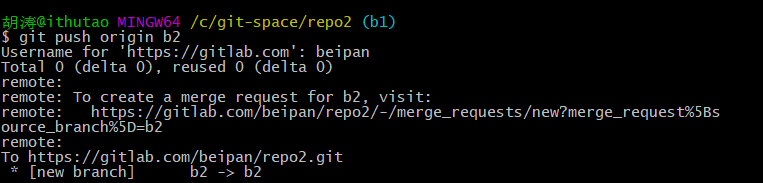


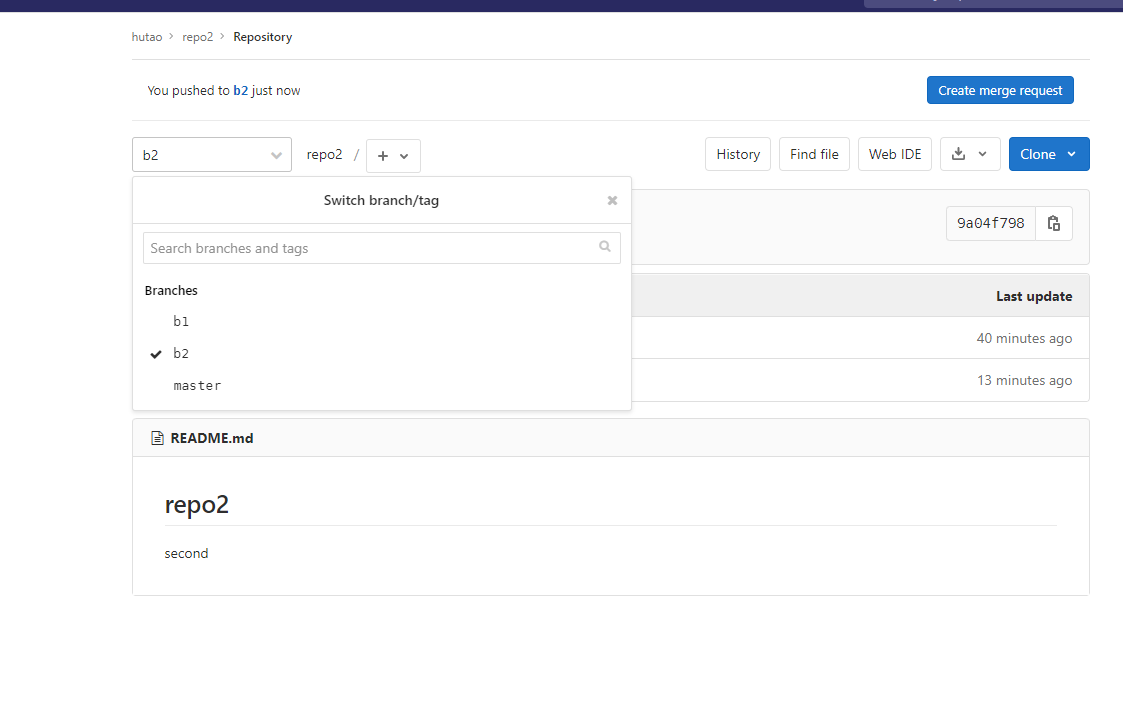
推送至远程仓库分支

git push origin b1

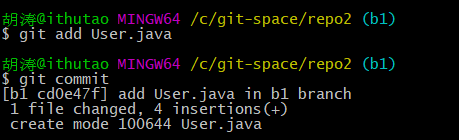


git push origin b2

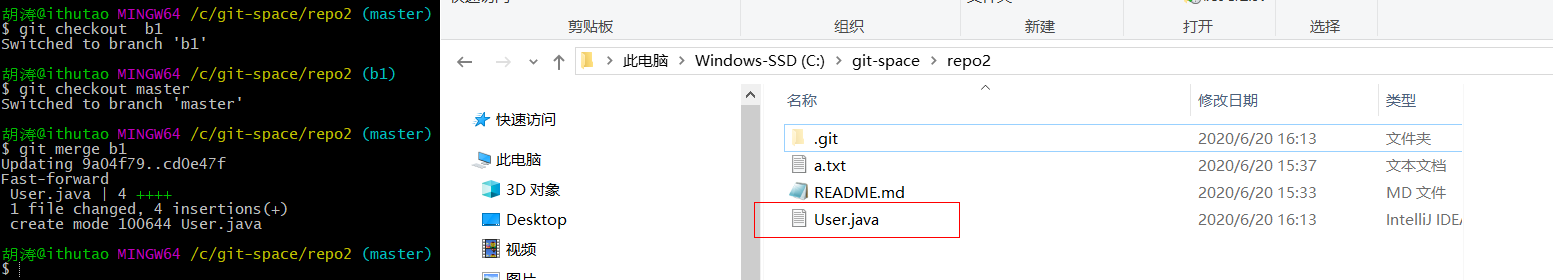




在b1中增加文件User.java 并添加到本地仓库

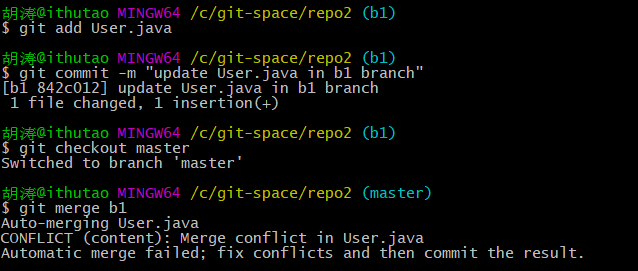


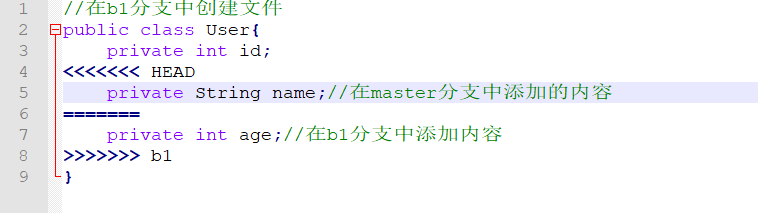
合并b1到master中

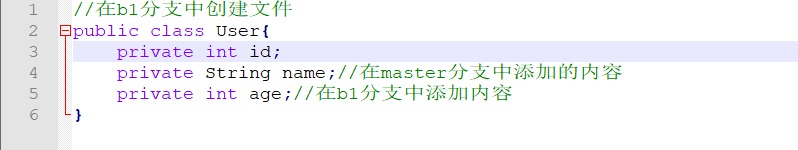


在b1中 修改User.java后 再合并到master中

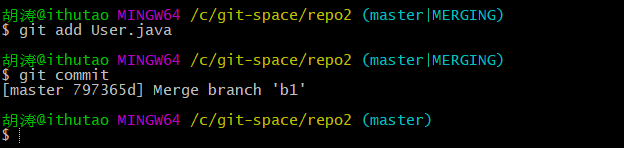
这时发生冲突



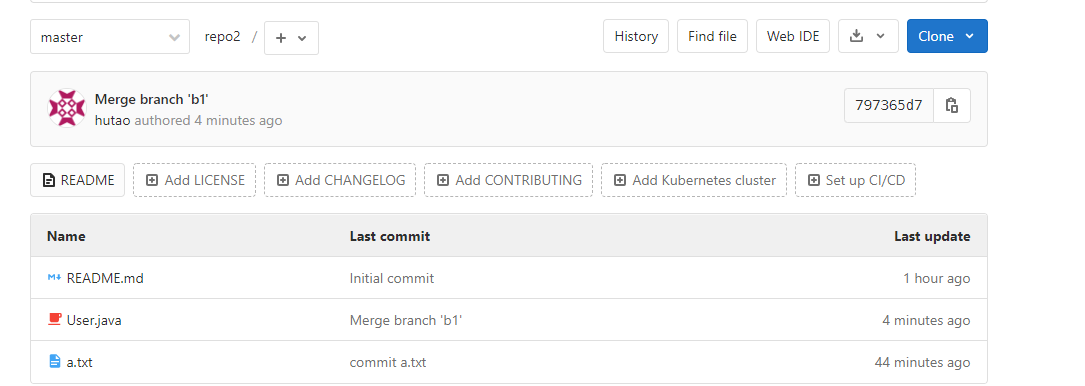


手动删除无用的标记：

重新提交



修改本地文件并解决冲突后推送到远程仓库



删除本地分支

git branch -d b1

删除远程分支

git push origin -d b1

创建标签

git tag v1.0

推送标签

git push origin v1.0

检出标签

git checkout -b b2 v2.0

删除本地tag

git tag -d v1.0

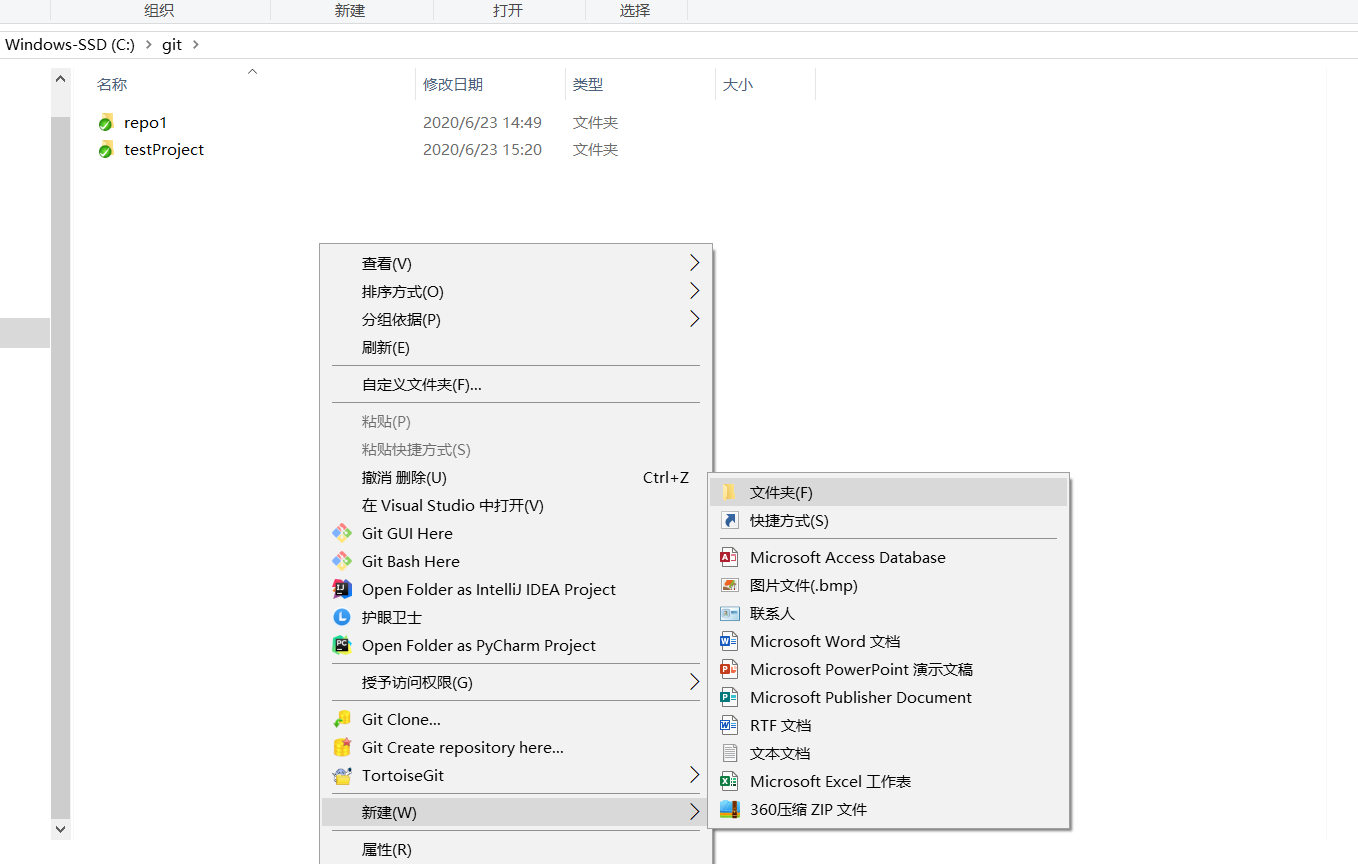
删除远程tag

git push origin3 :ref/tags/v1.0

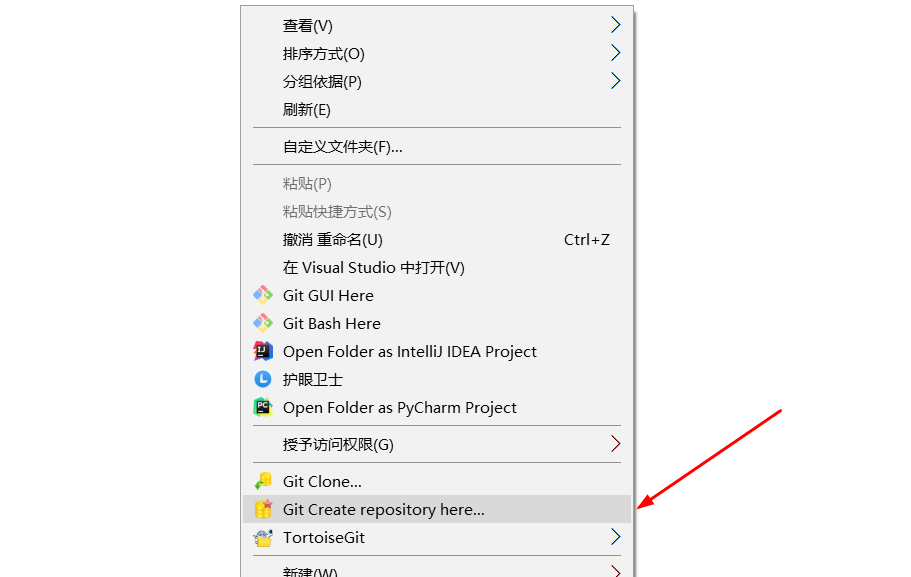
# TortoiseGit使用教程

## 创建本地空仓库

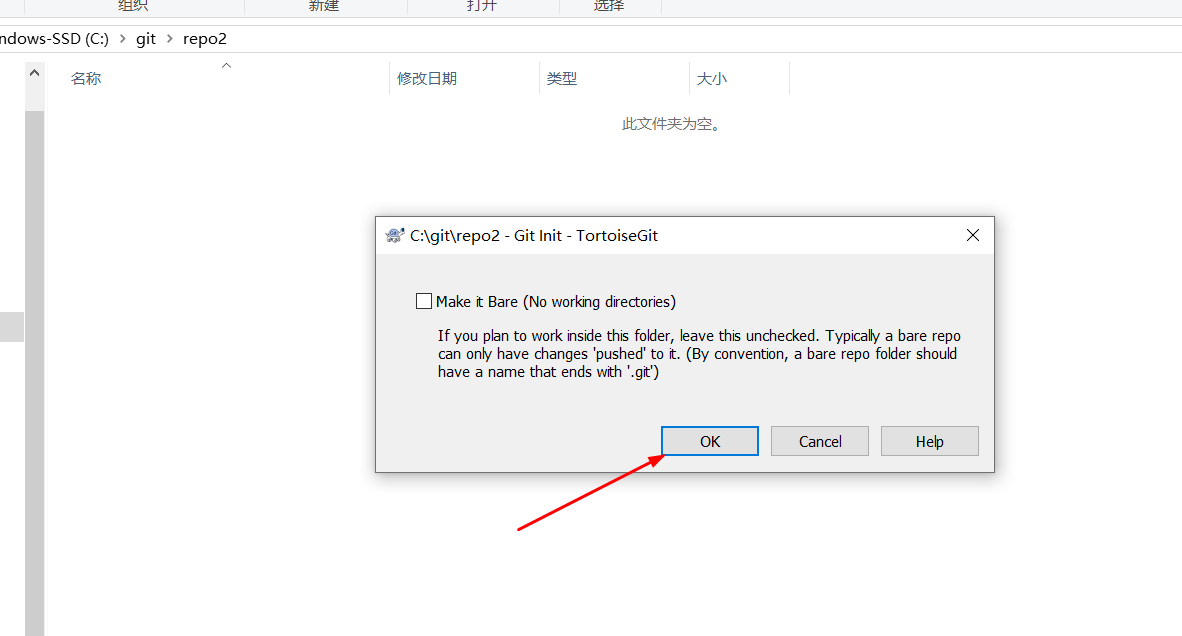
1 首先创建一个空的文件夹 命名为repo2



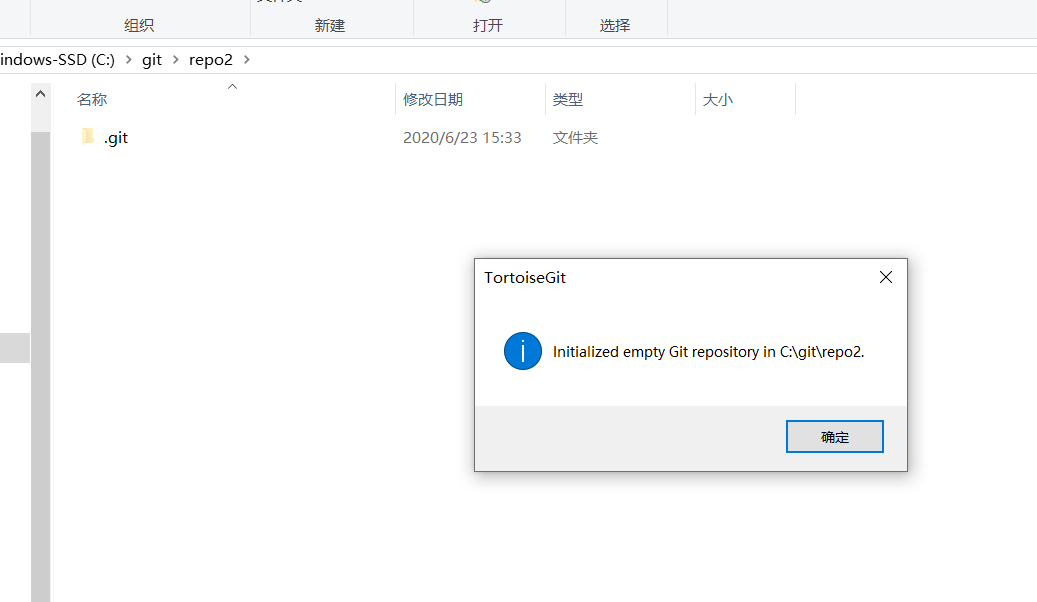
2进入该文件夹 右键



3直接点击OK （点击OK即可立即初始化一个仓库）

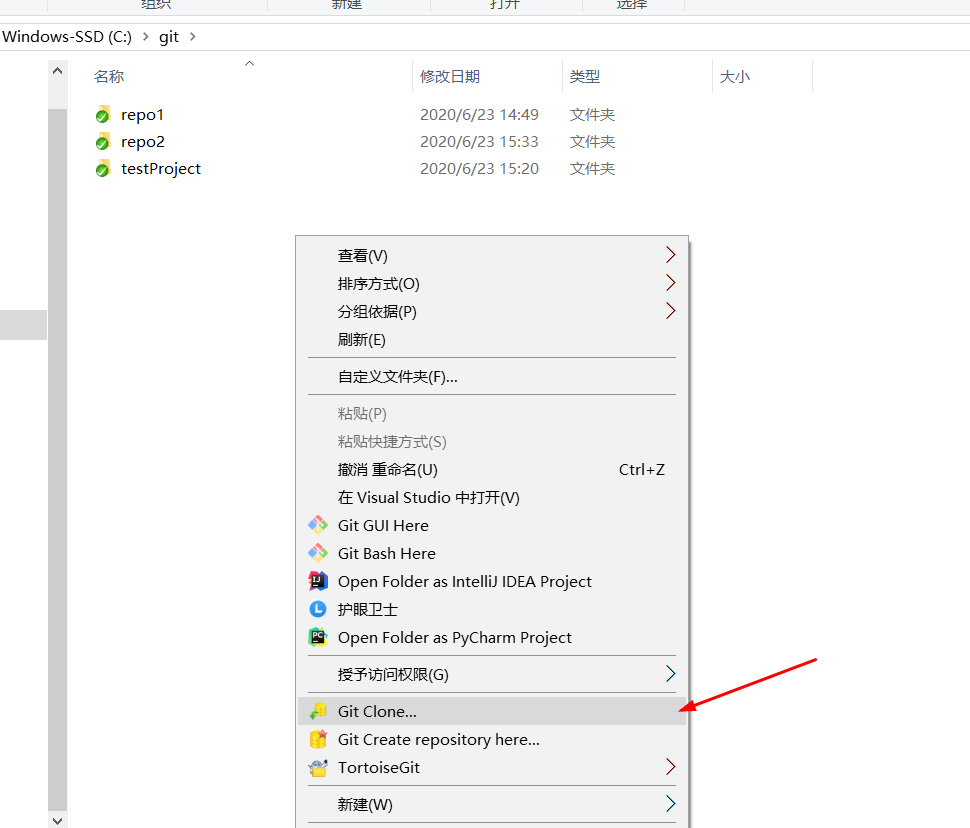


4点确定

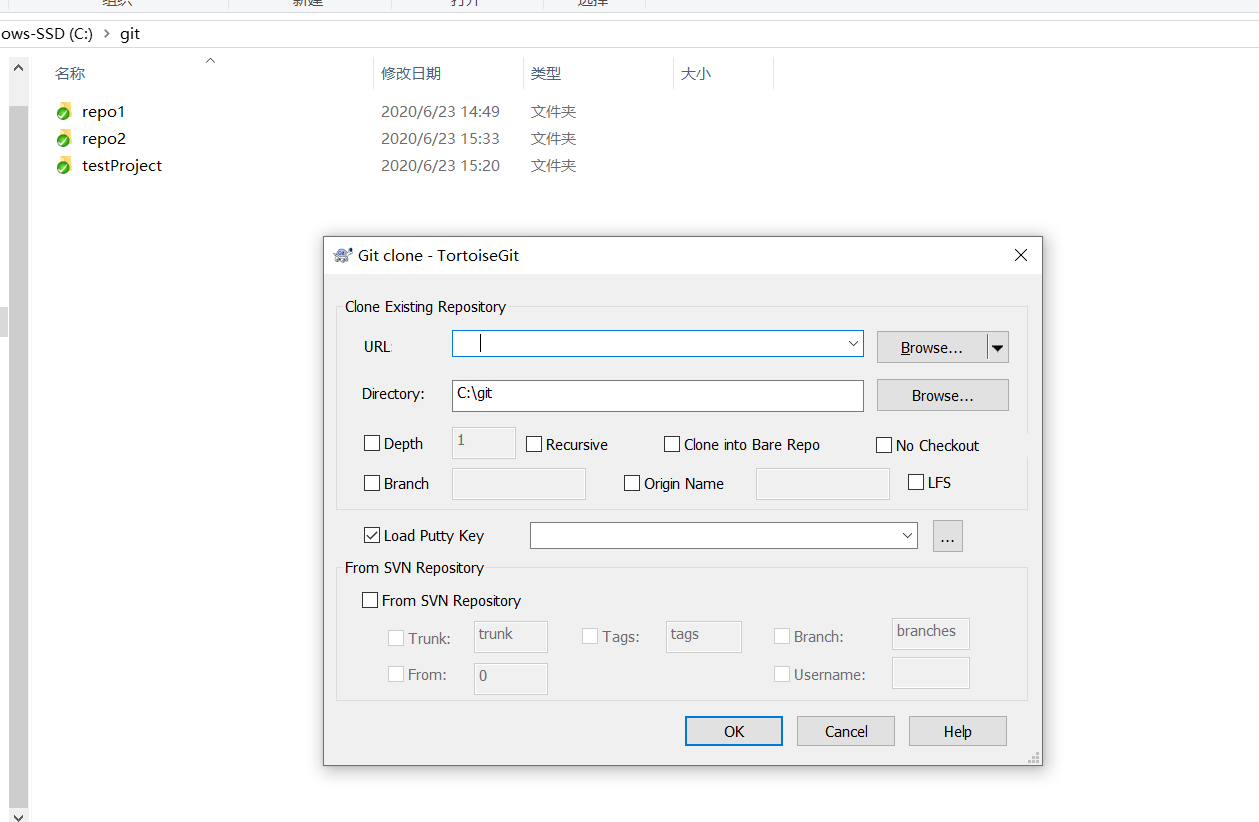


## 克隆远程仓库到本地

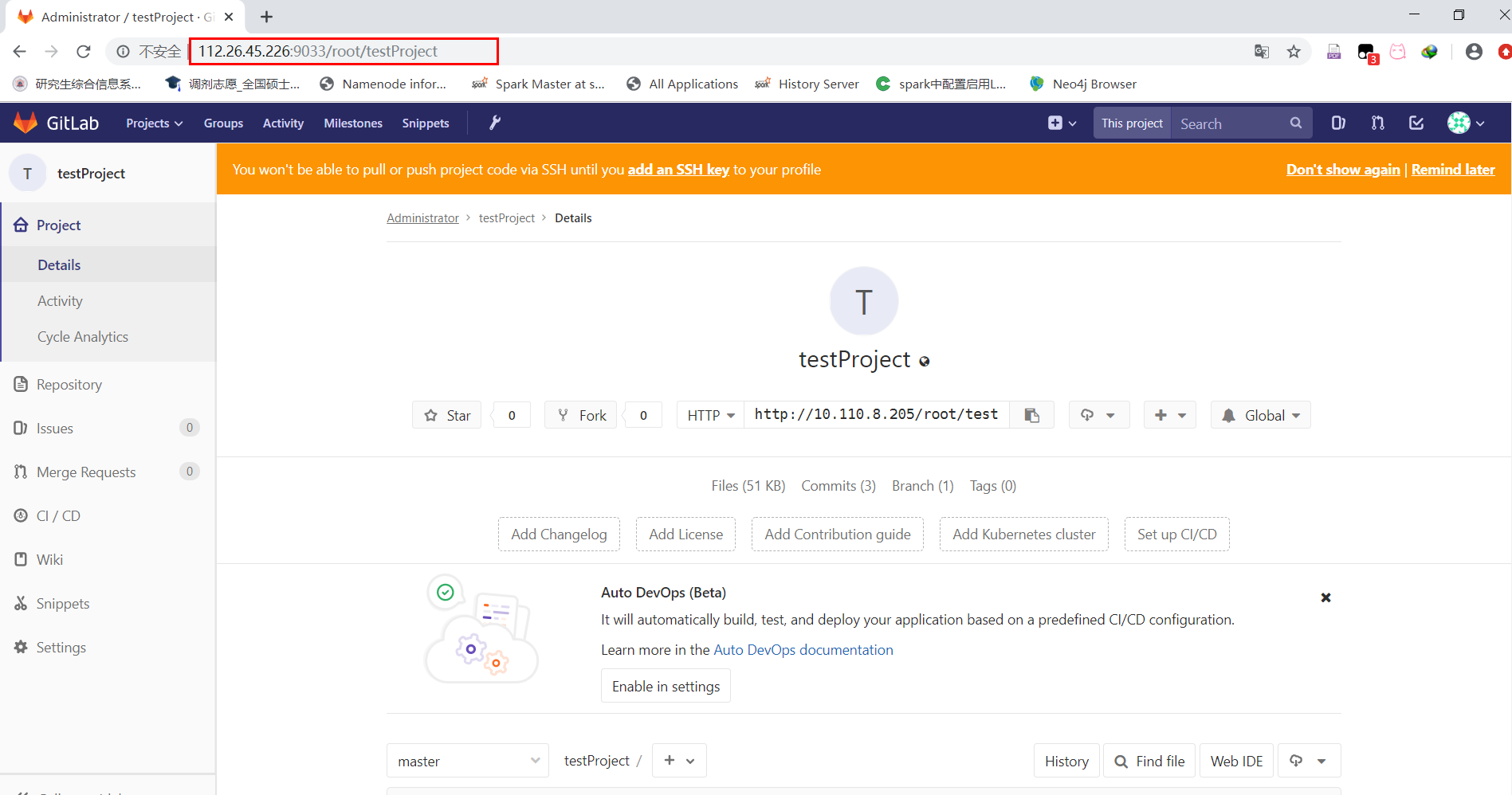
1右键

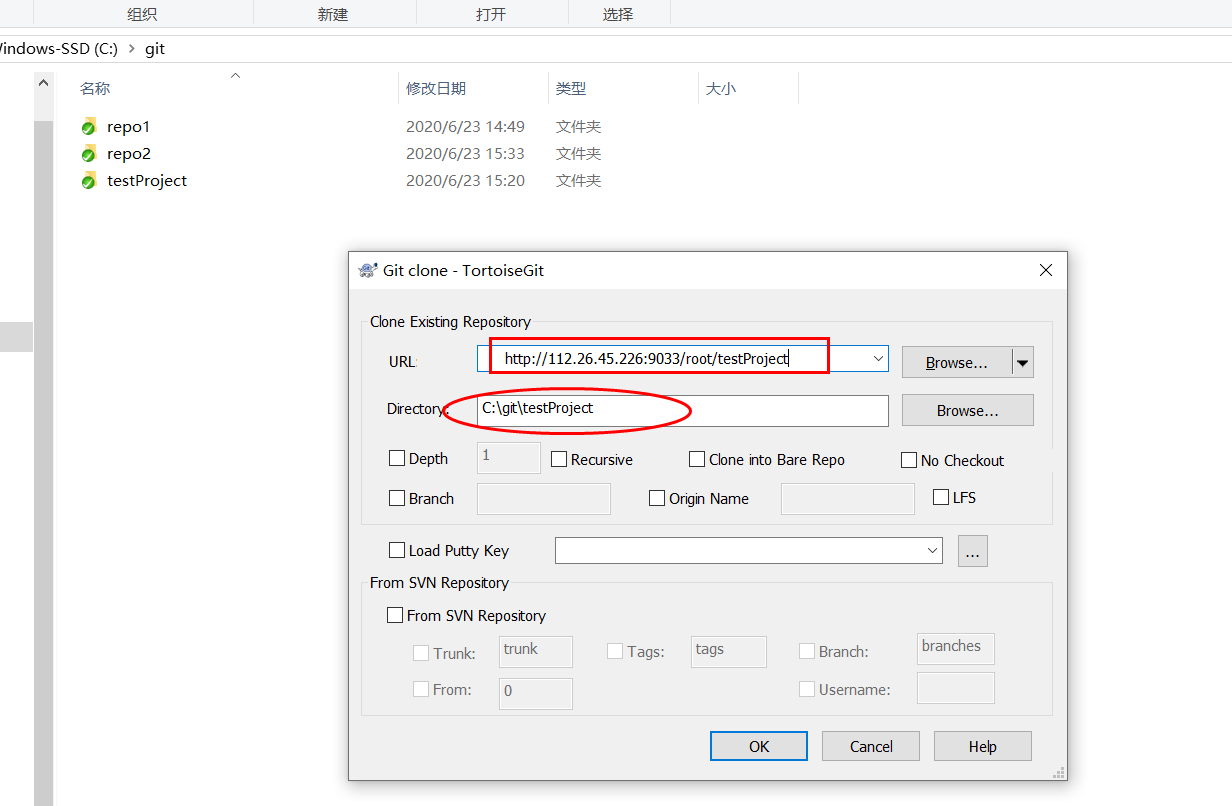


2 出现该界面 需要输入地址

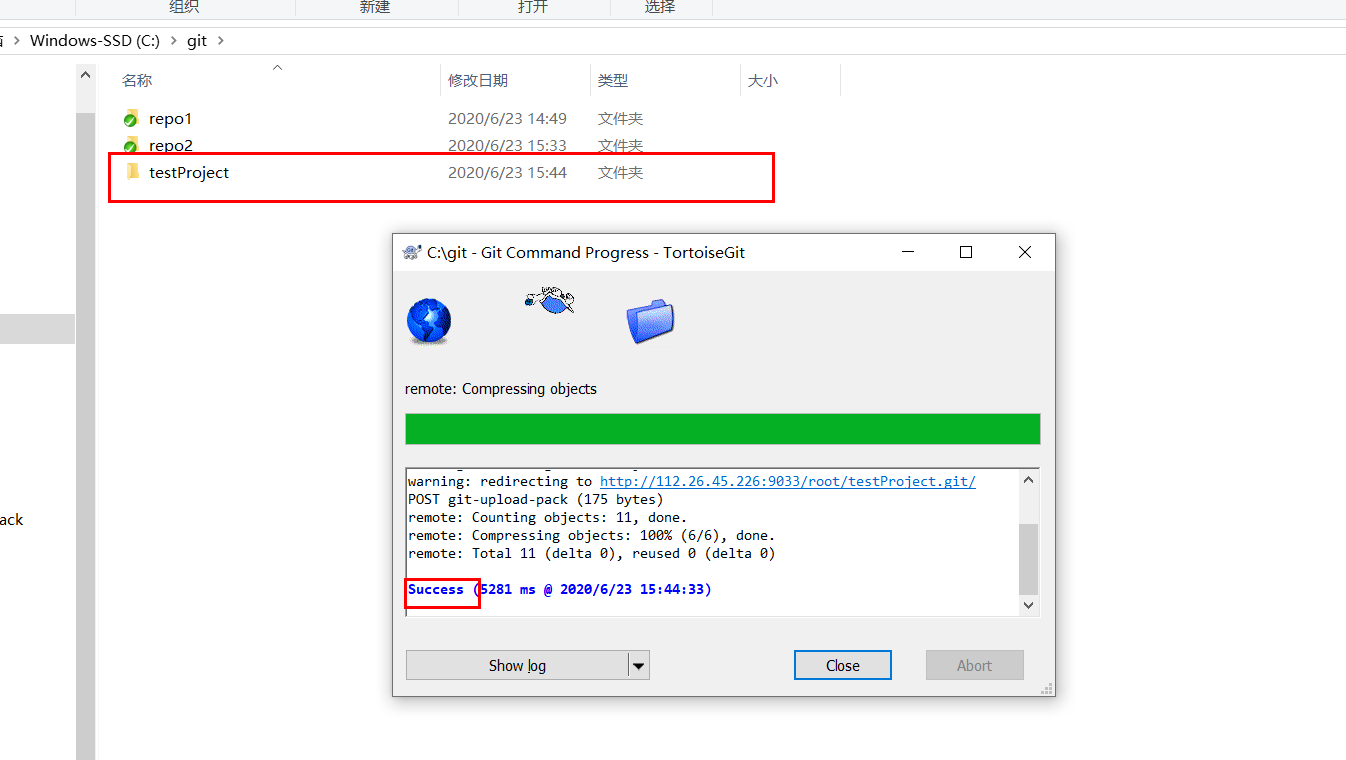


3 去网页登录自己的gitlab账户后 复制网址



4 粘贴到下图矩形位置（椭圆处呈现的为自动填充 表示本地仓库存放路径）然后点击OK

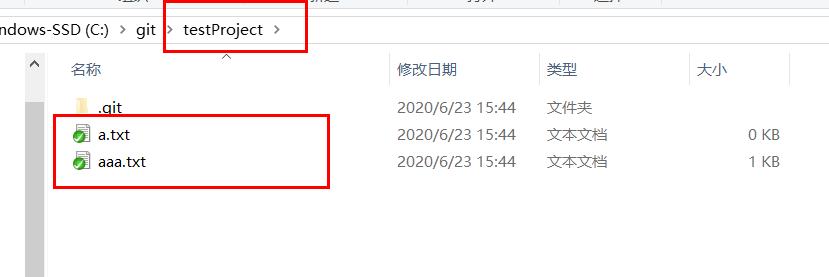
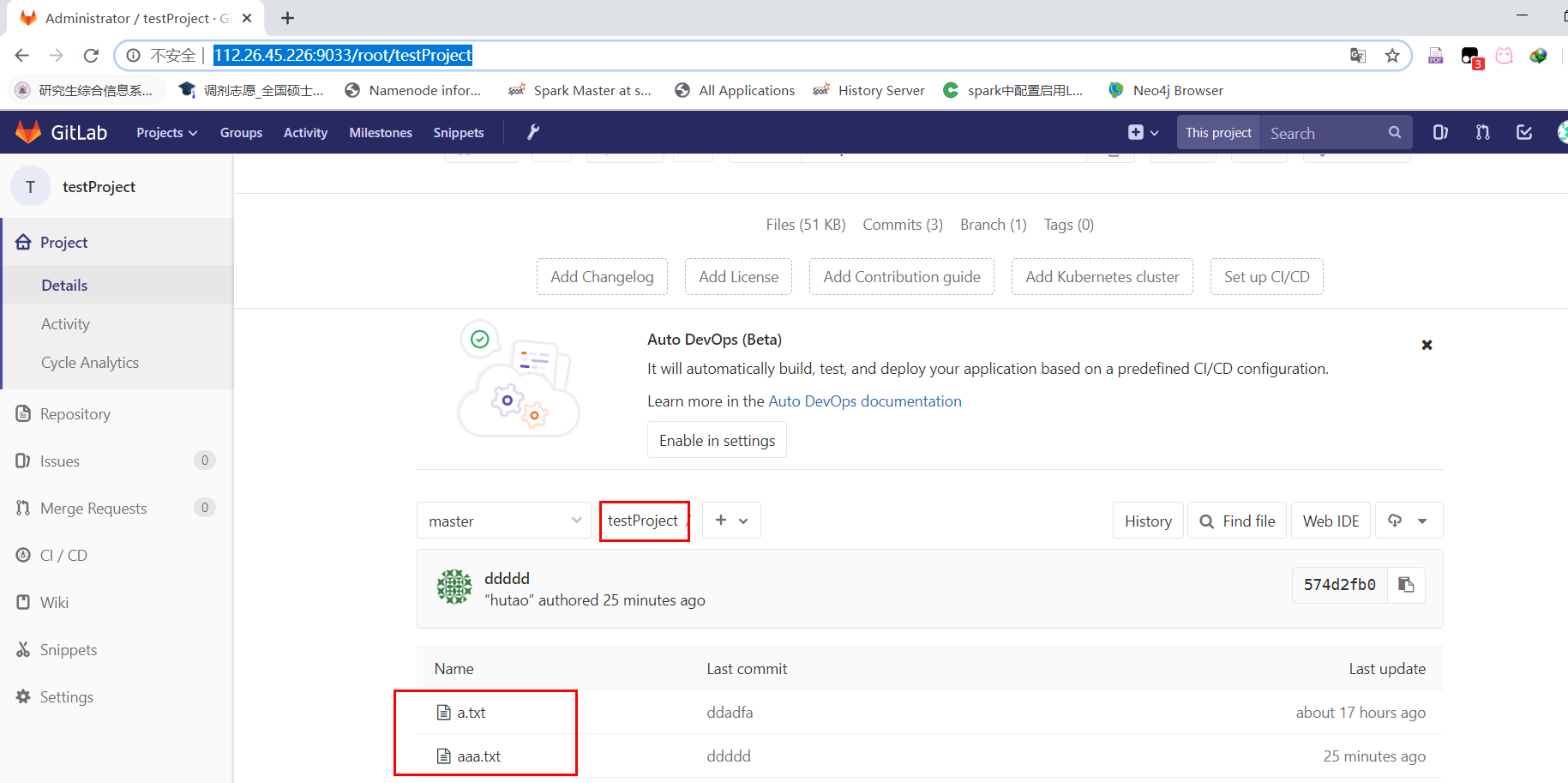
5 发现生成一个新文件夹 名字叫testProject（名字与远程仓库名相同） 点击close



6 发现图标 不一样与原有的两个仓库图标不一样 （反应有点慢）需右键刷新几次

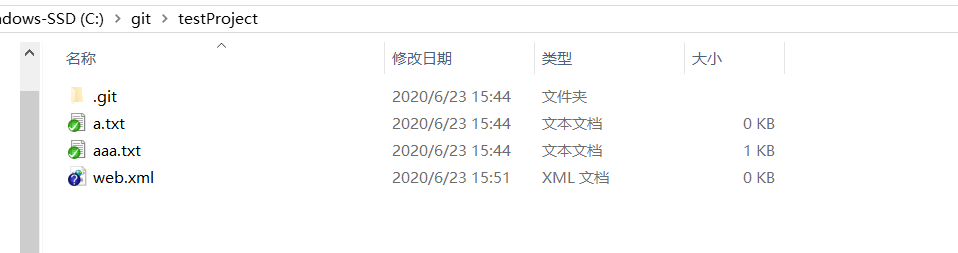


7 进入该仓库 发现其包含内容 与远程仓库内容一致

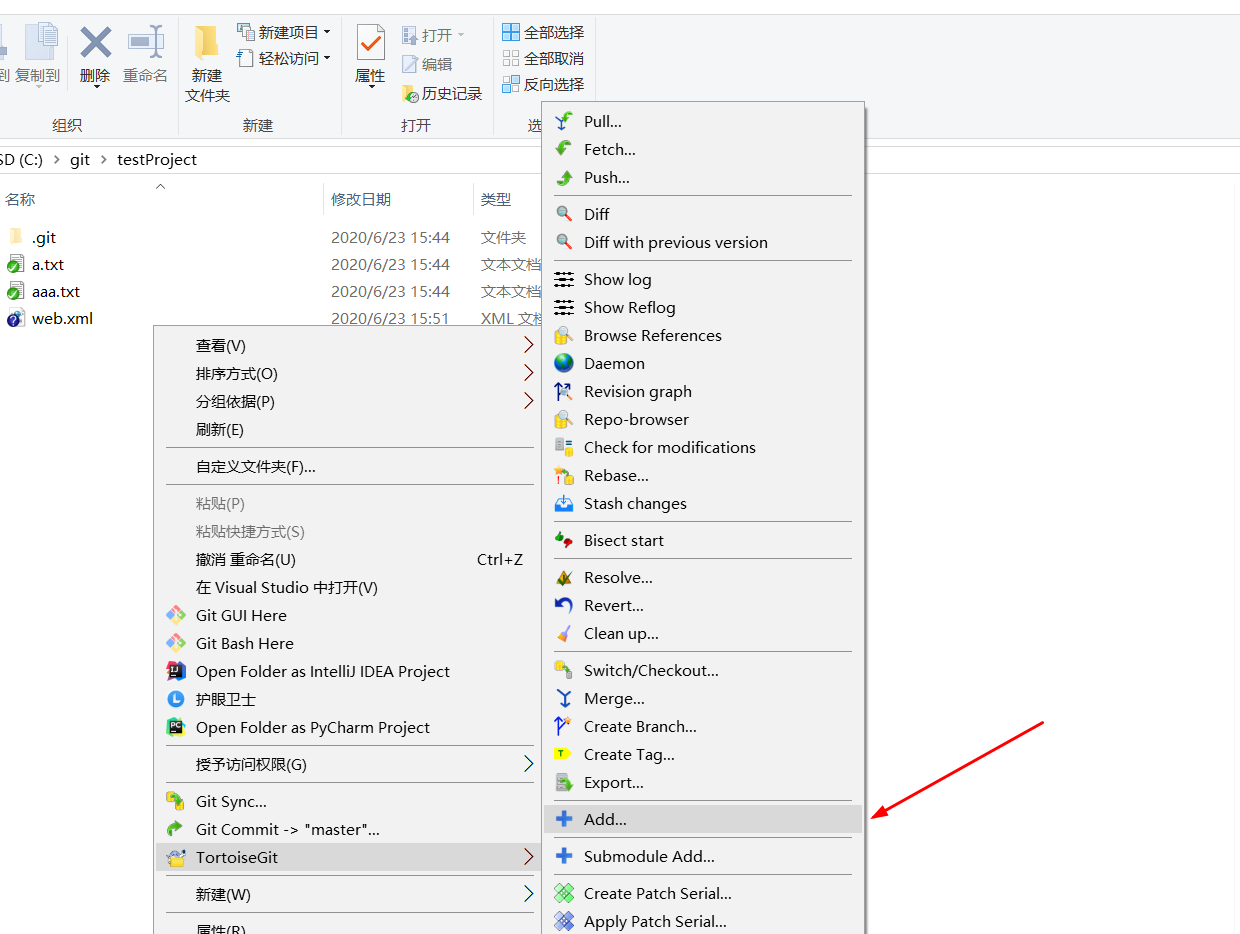
 

## 添加到暂存区并提交文件

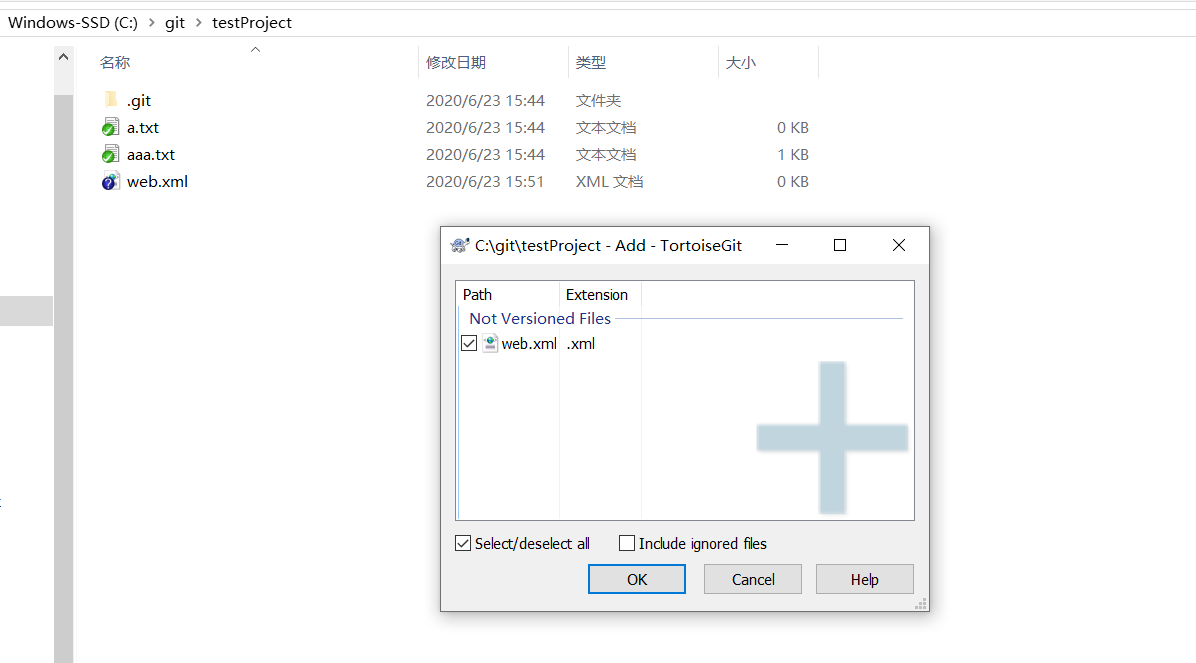
1 在本地仓库中 新建一个web.xml文件



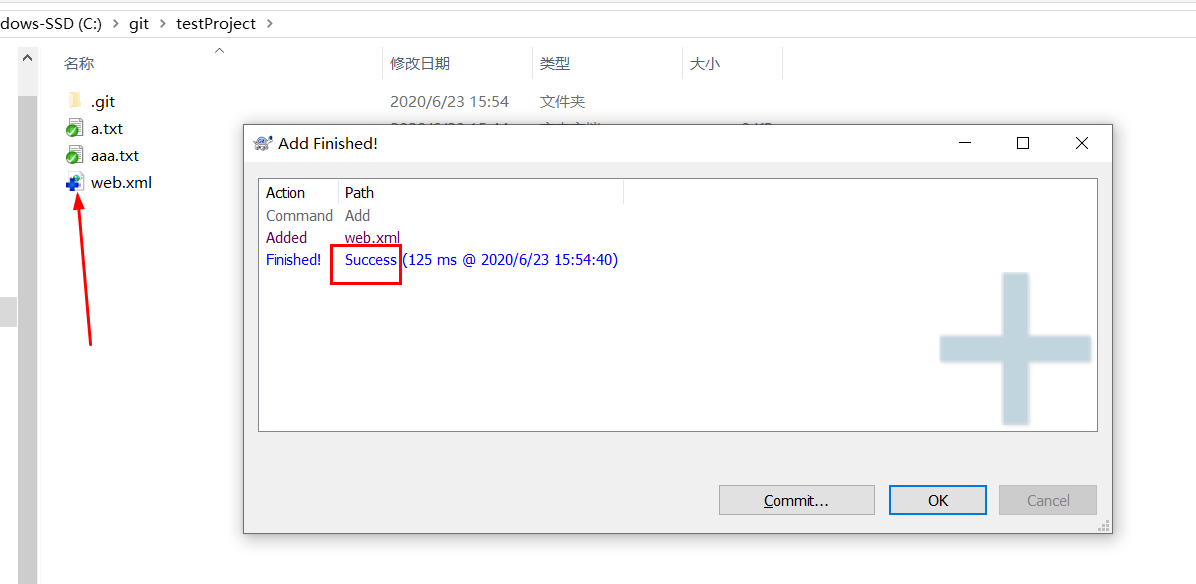
2 右击该文件 点击add



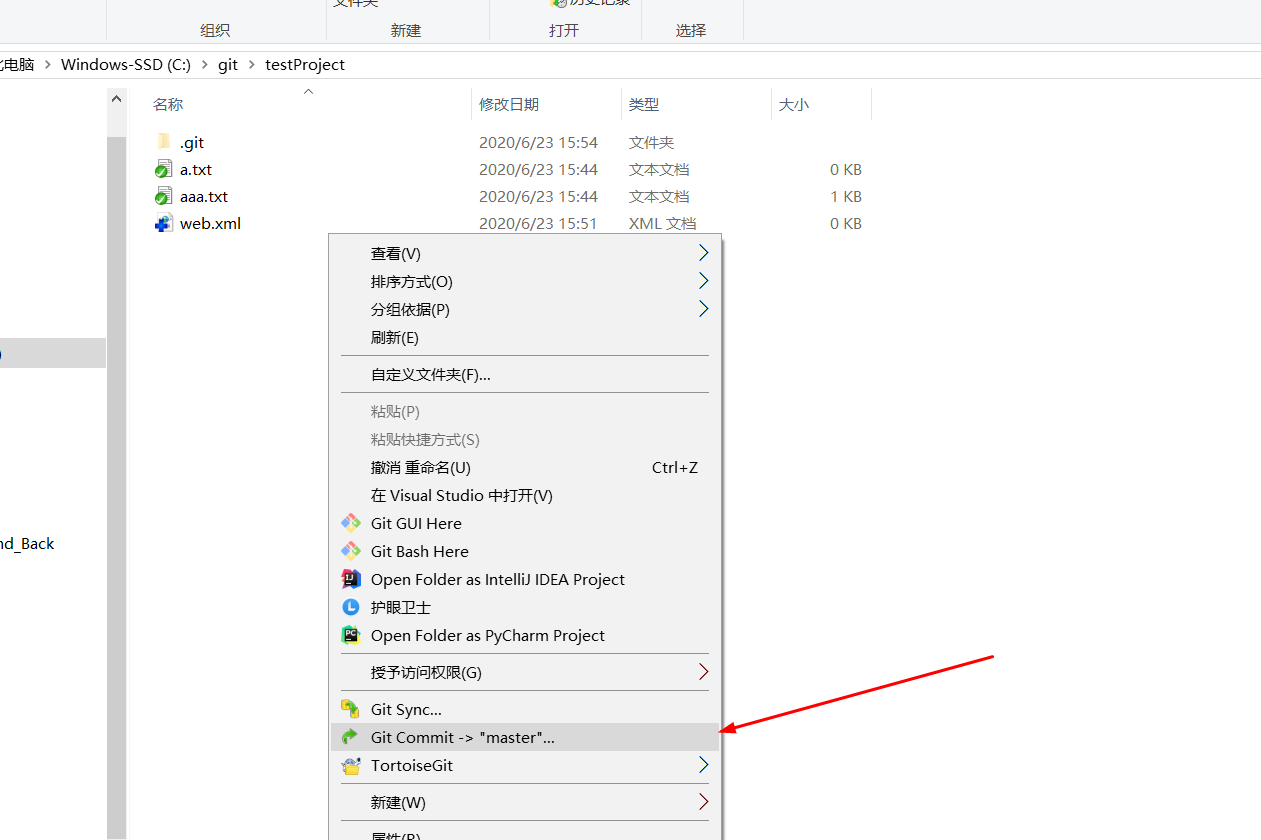
3 直接点击OK



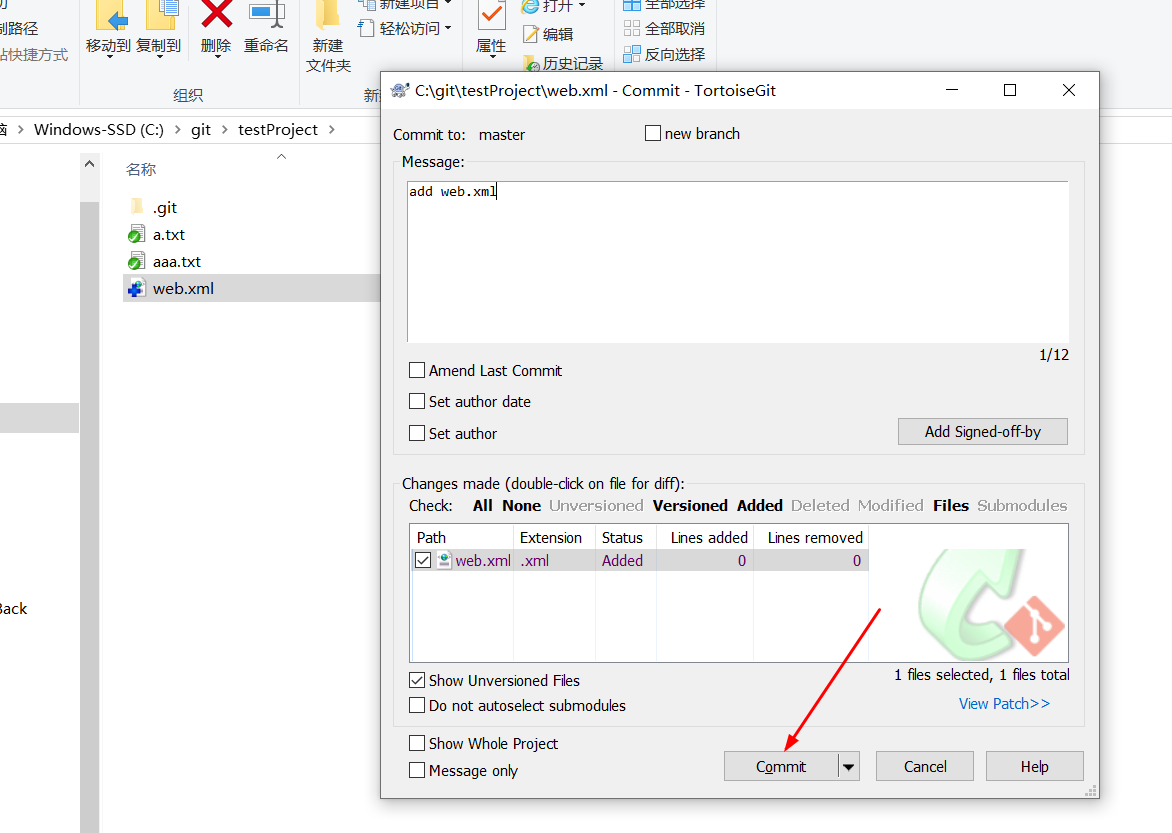
4 点击OK后出现下图 web.xml 已被添加到暂存区 此时可以点OK按钮 也可以点击 Commit按钮 我这里点击OK按钮



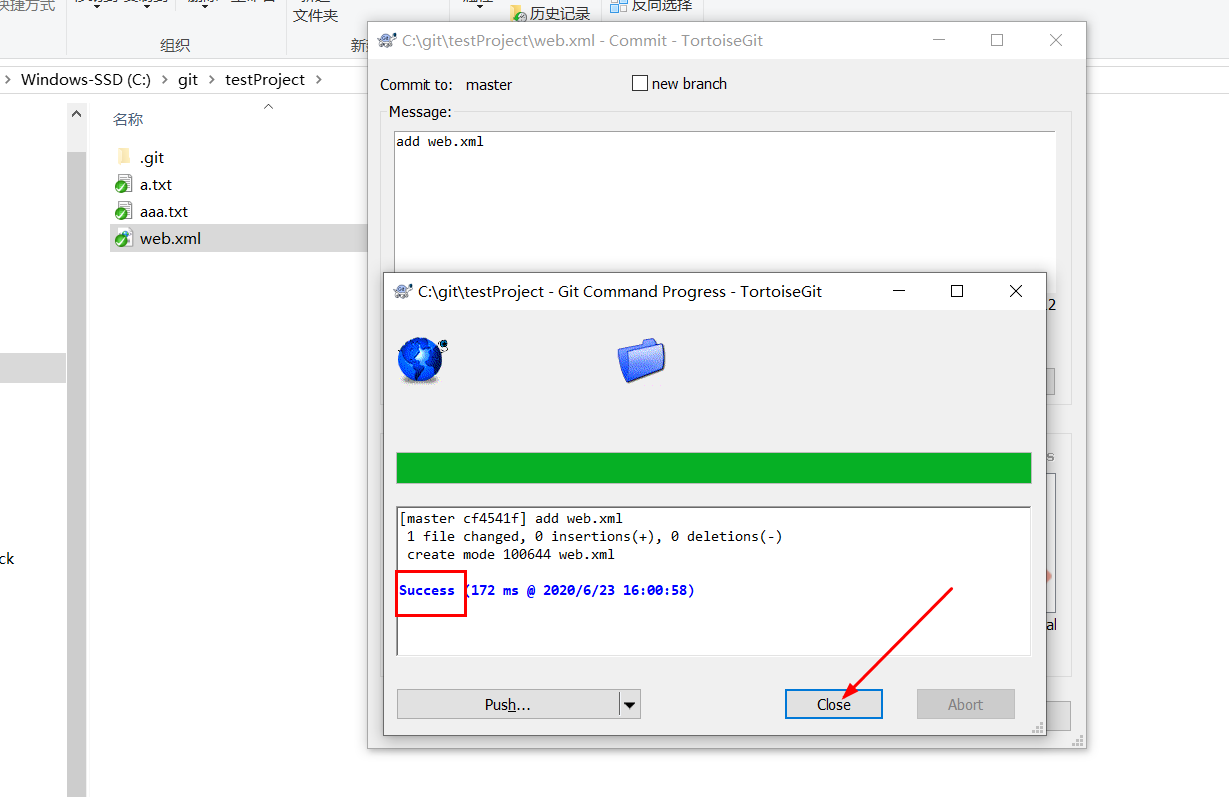
5 右击该文件 进行commit



6 填写一下信息（此处必须填写！！！） 表示进行具体提交什么文件的操作 然后点击Commit



7点击Close

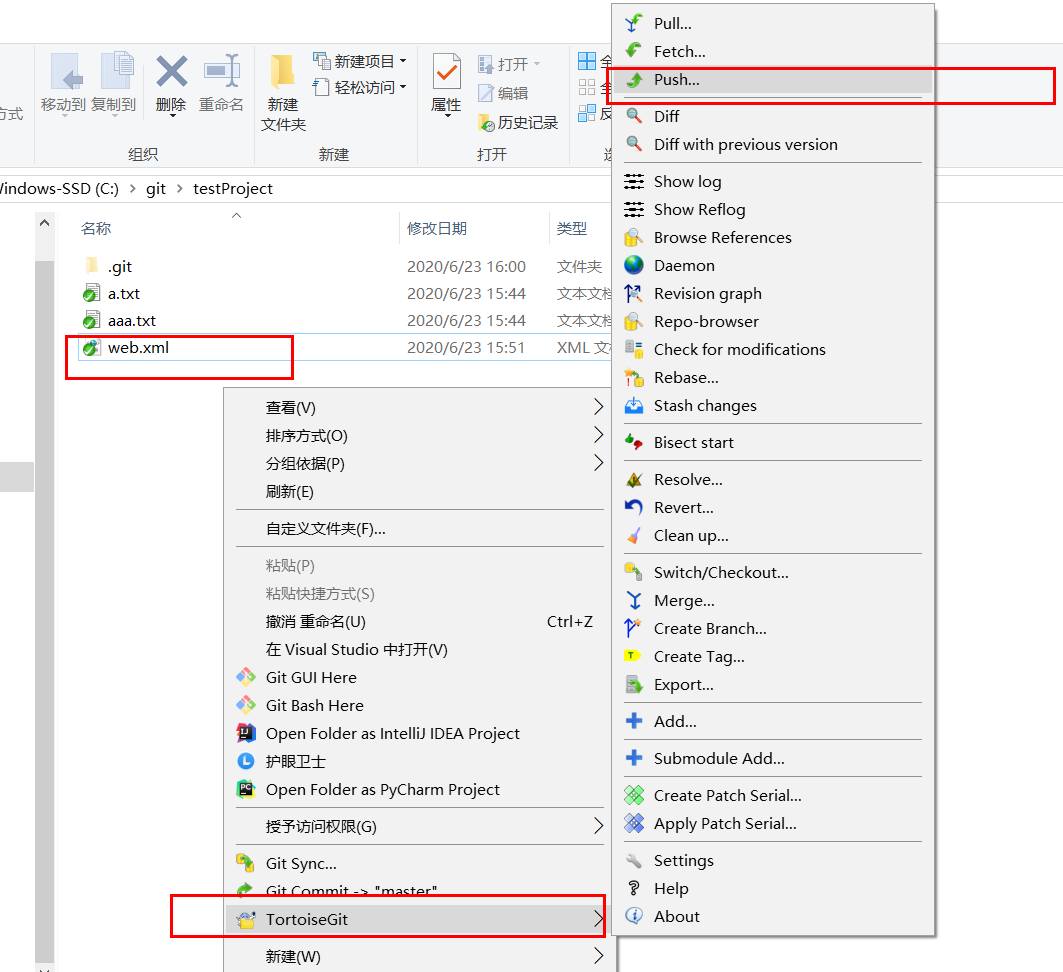


## 推送本地仓库到远程仓库1

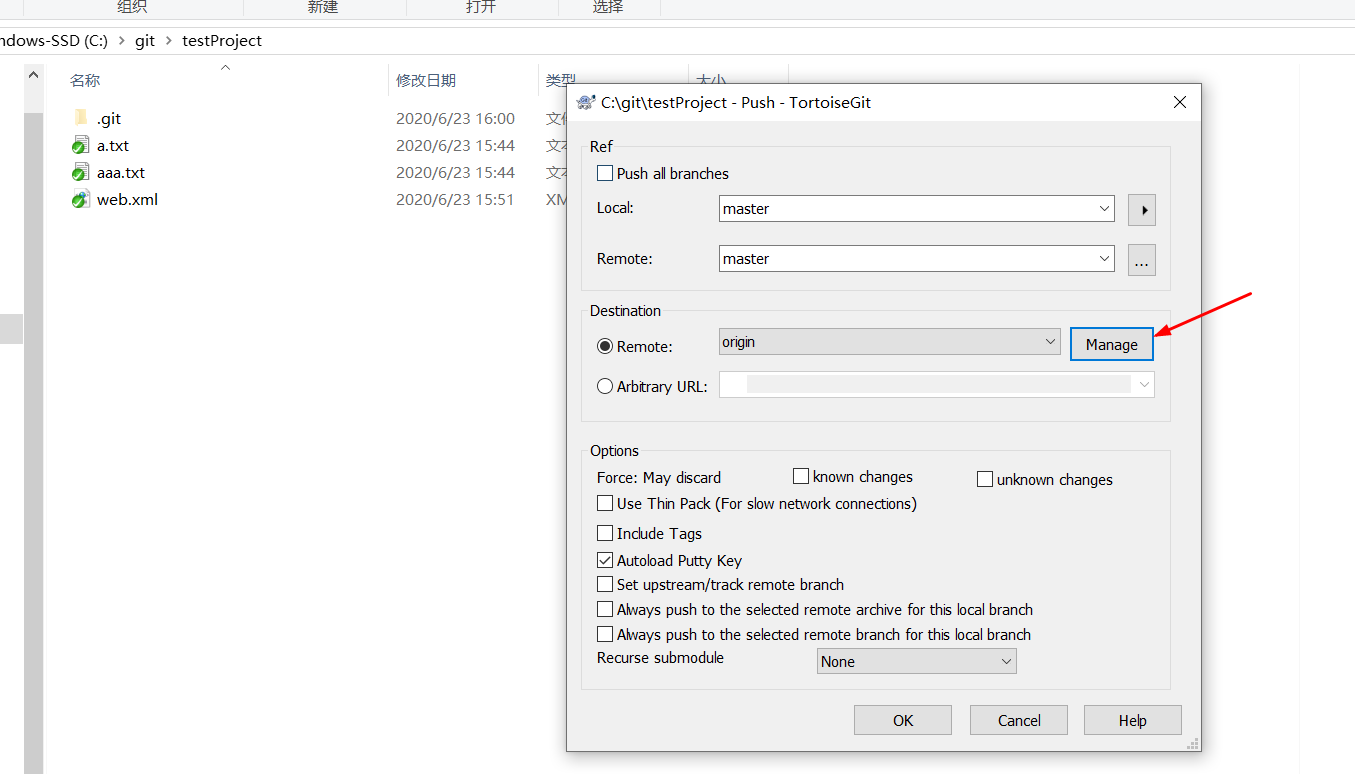
(推送仓库分两种情况 注意：testProject仓库是之前从远程仓库克隆下来的)

情形1 推送从远程克隆下来的仓库（简单情形）

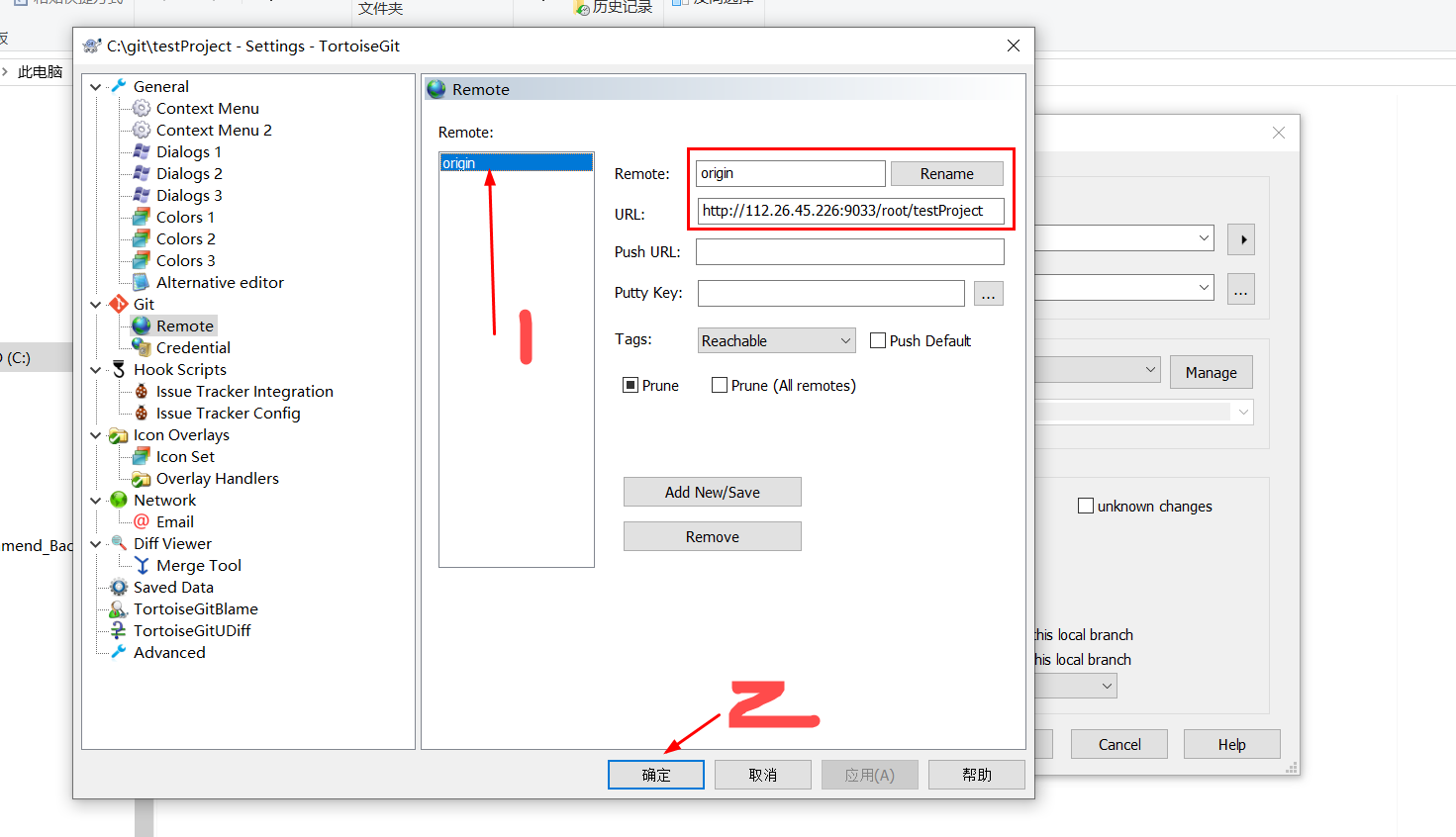
1由于上文已经有了修改 新增web.xml文件 将这次修改推送到远程仓库



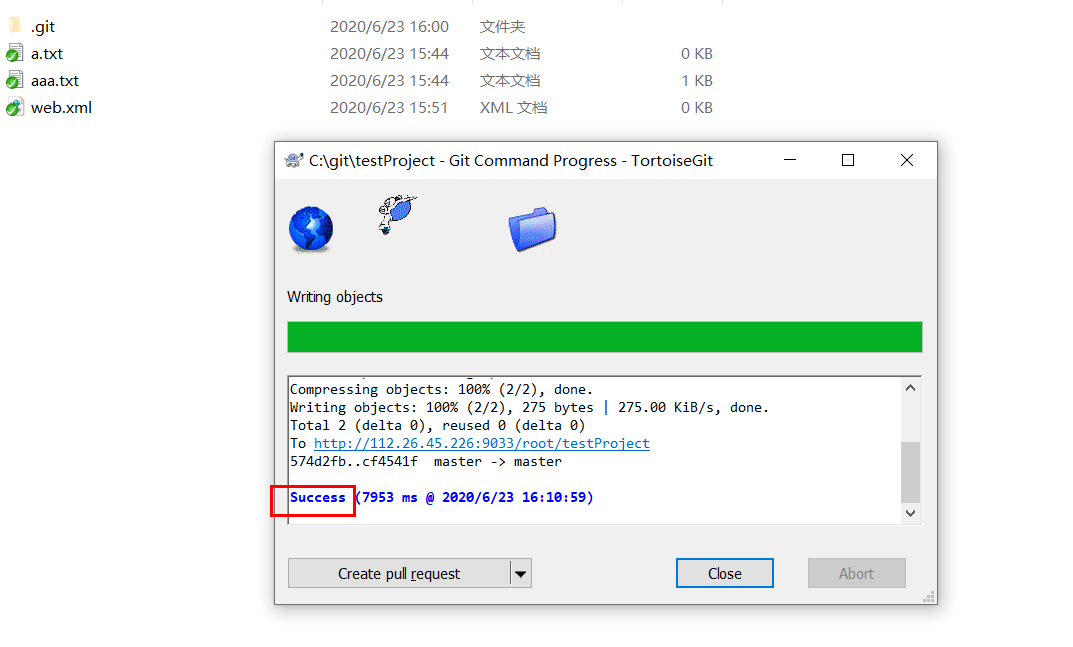
2出现下图 不妨点击一下Manage按钮



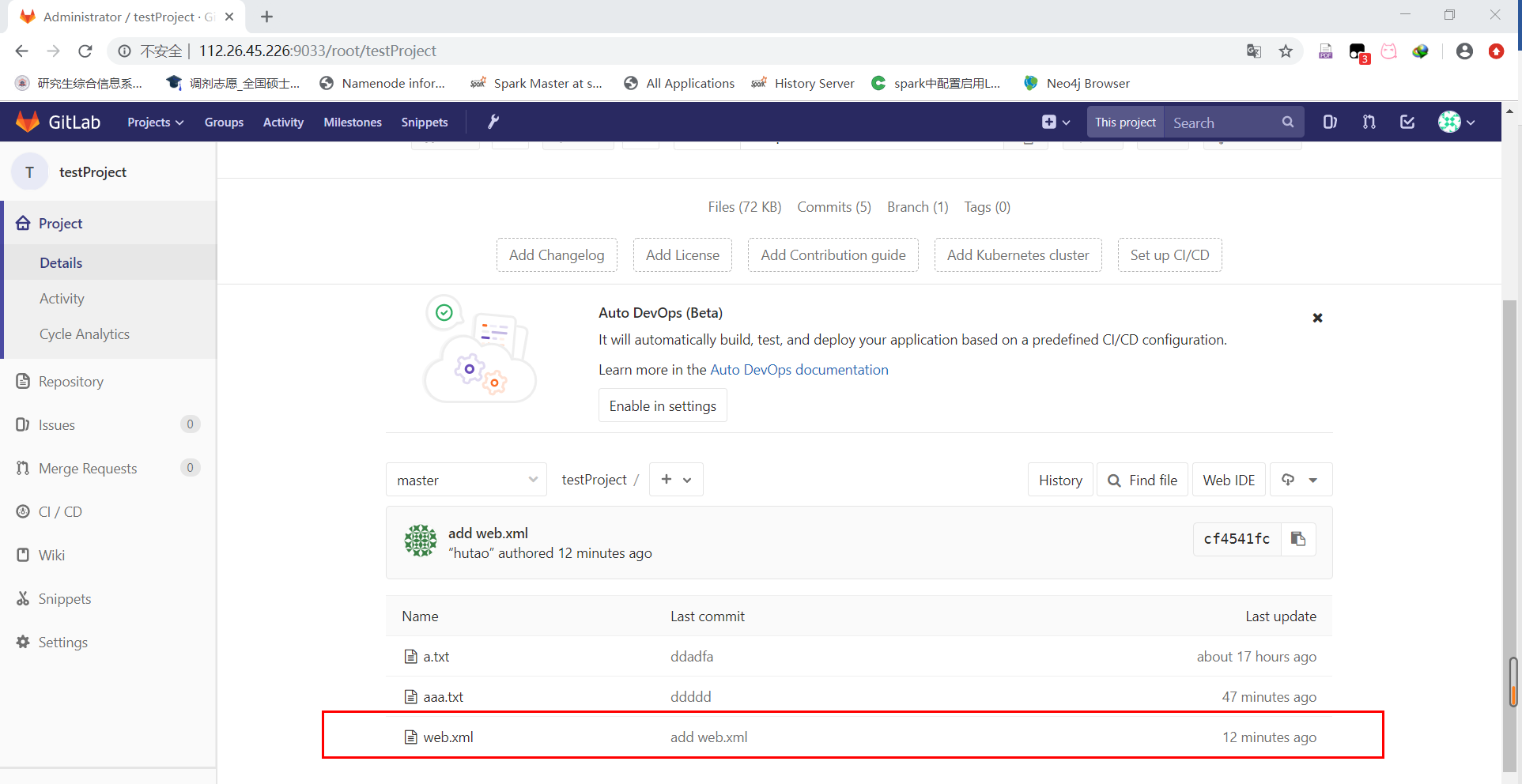
3 在下图手动点击origin 便可出现矩形框处内容 即为远程仓库地址 然后点击确定



4 推送成功 点击Close



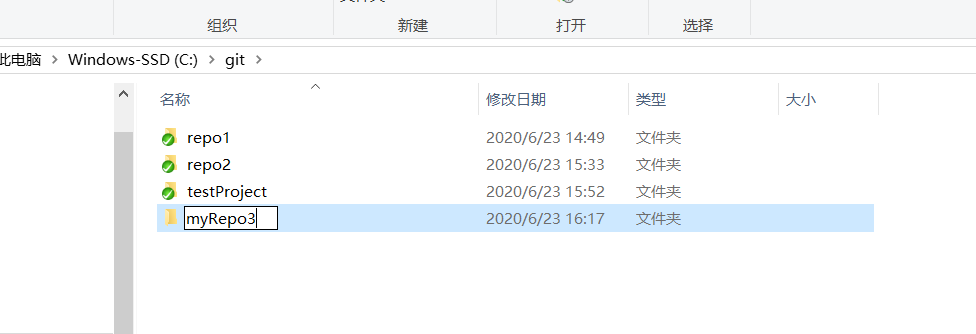
5 到网页刷新一下网址 发现 已经成功添加到远程仓库



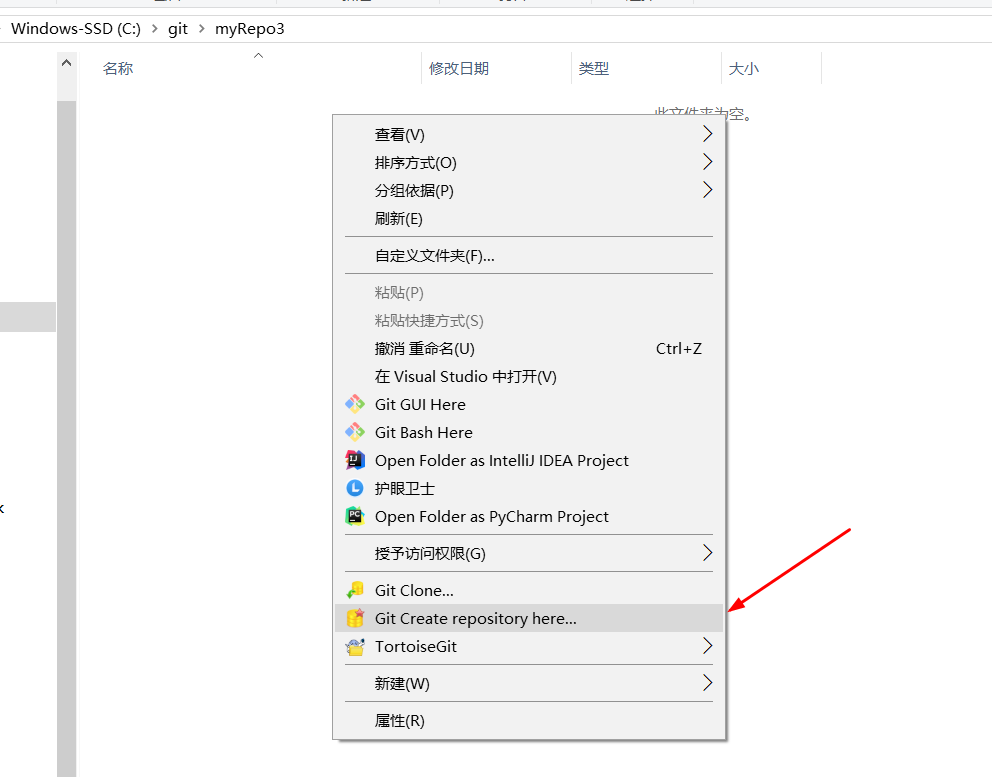
## 推送本地仓库到远程仓库2

情形2 推送本地自己创建的仓库（复杂情形）

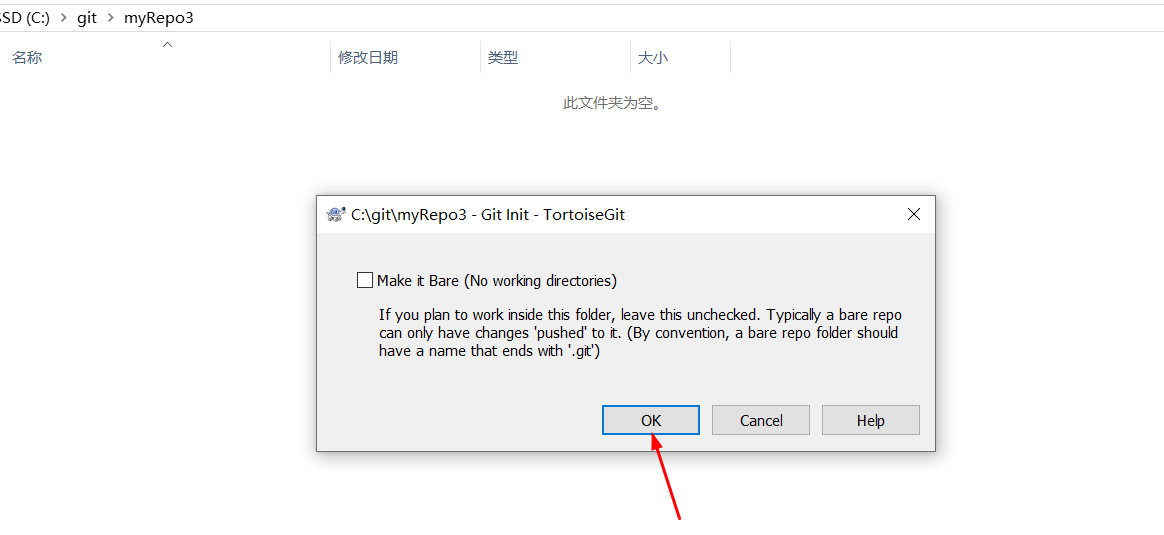
1 创建一个空文件夹



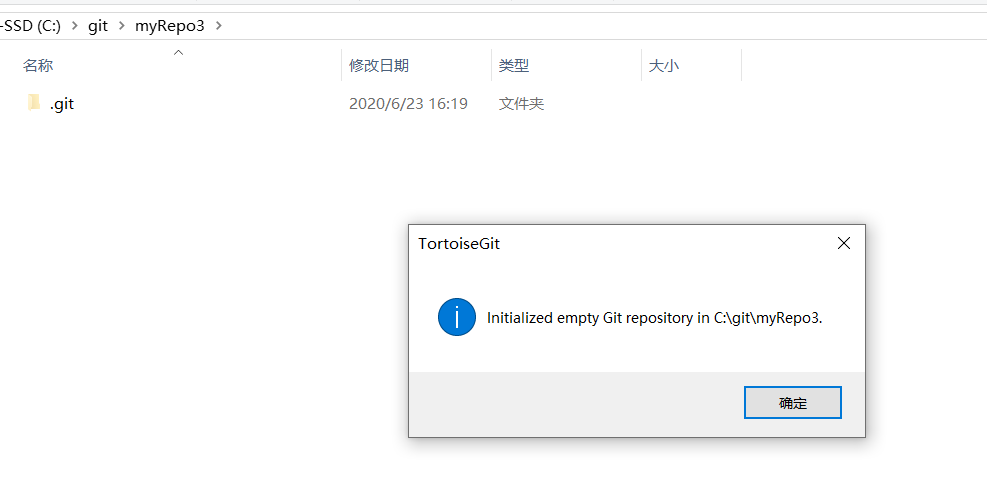
2 进入该文件夹 右键



3 点OK



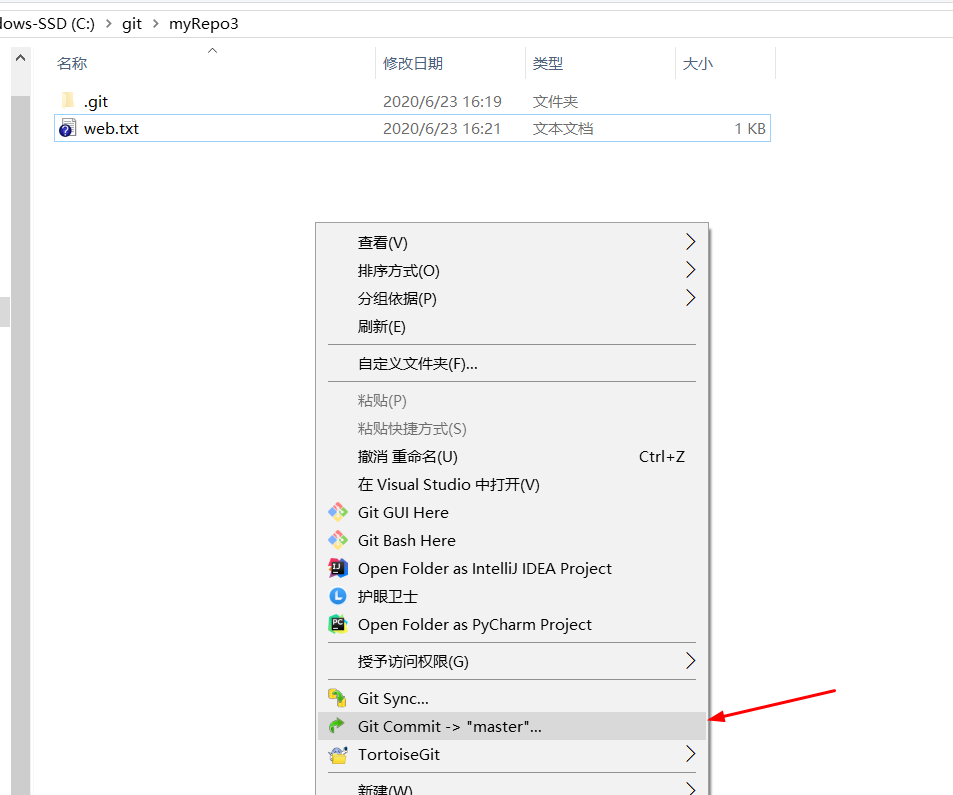
4 点击确定



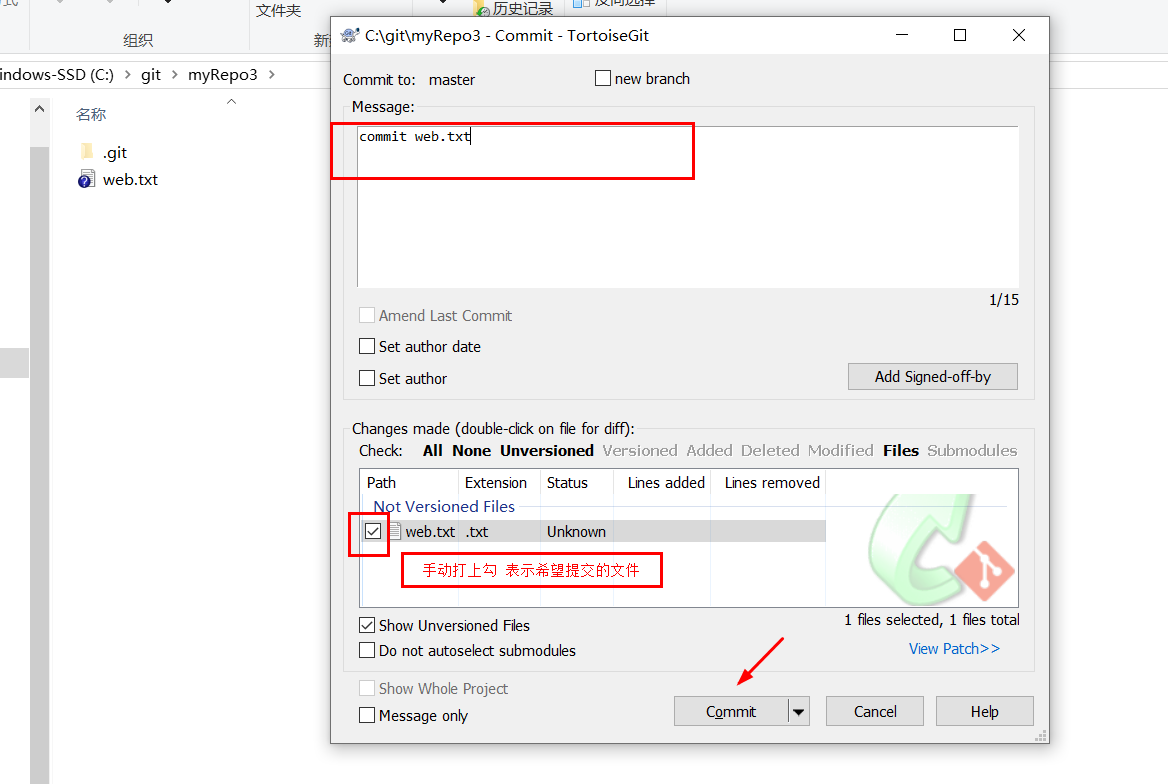
5 新建一个文件 自定义内容 如下 表示自己的开发代码



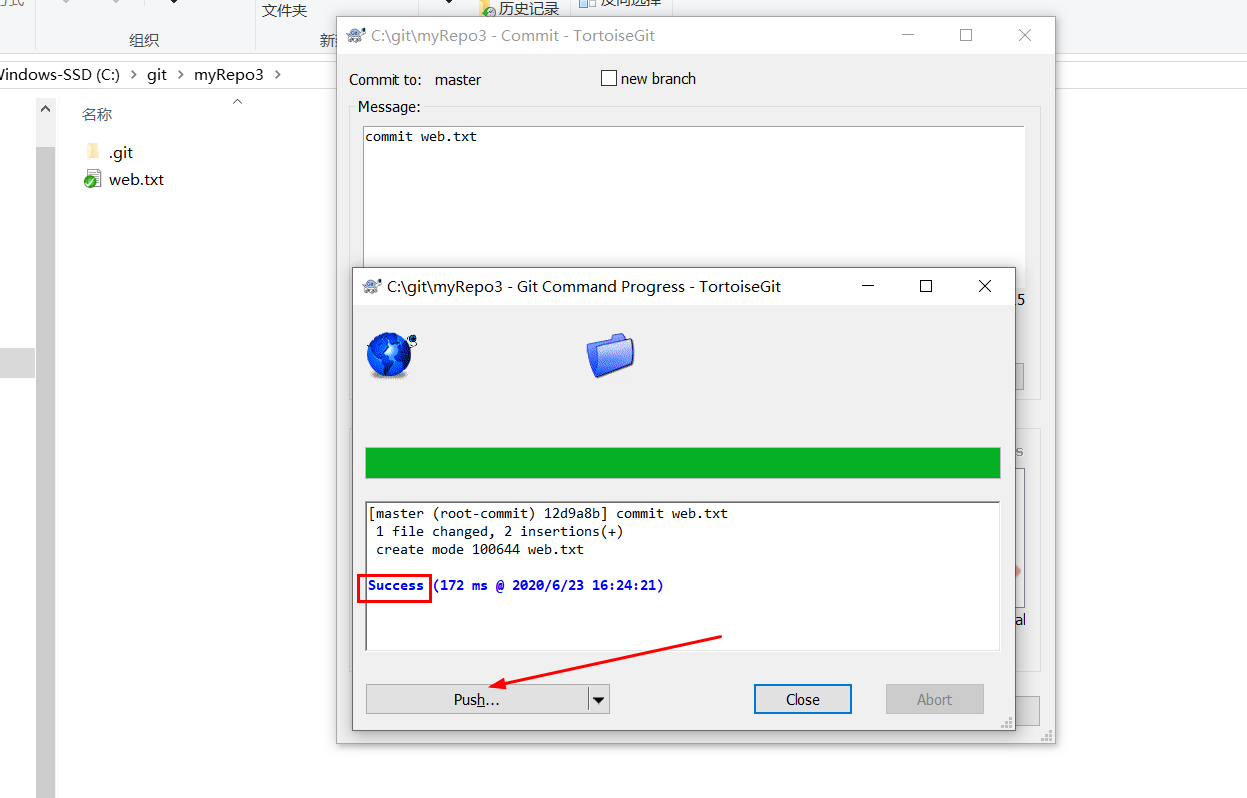
6 可以直接提交到本地仓库



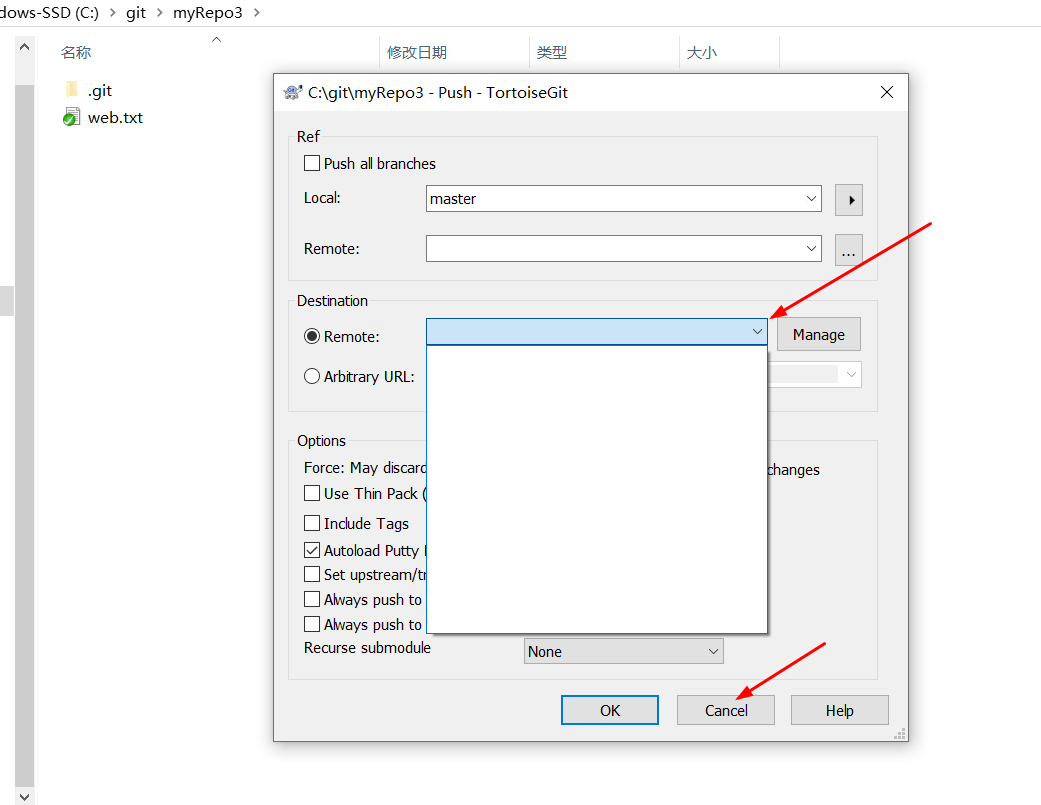
7 输入信息 打勾后 提交



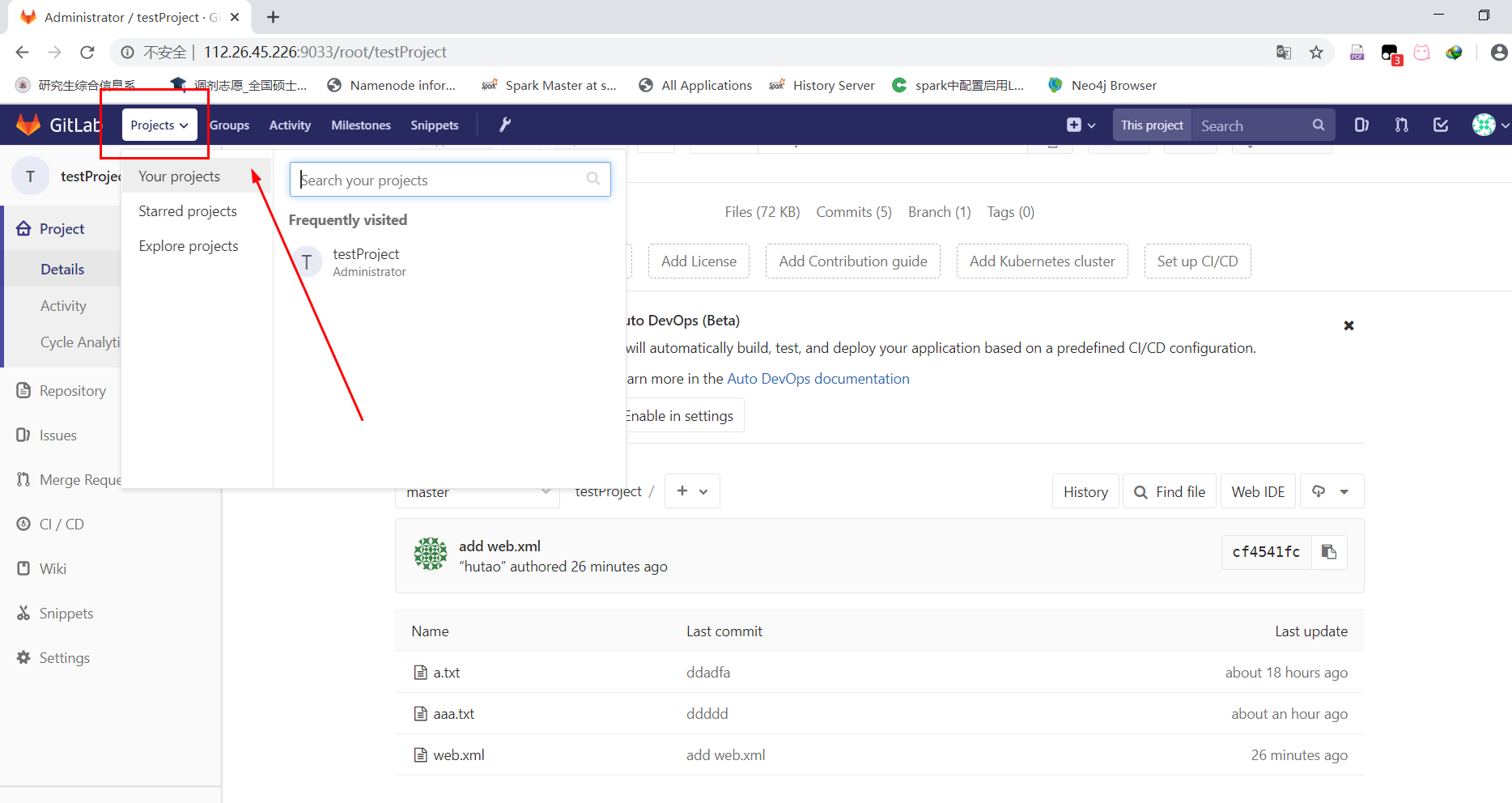
8 这里点击Push 进行推送



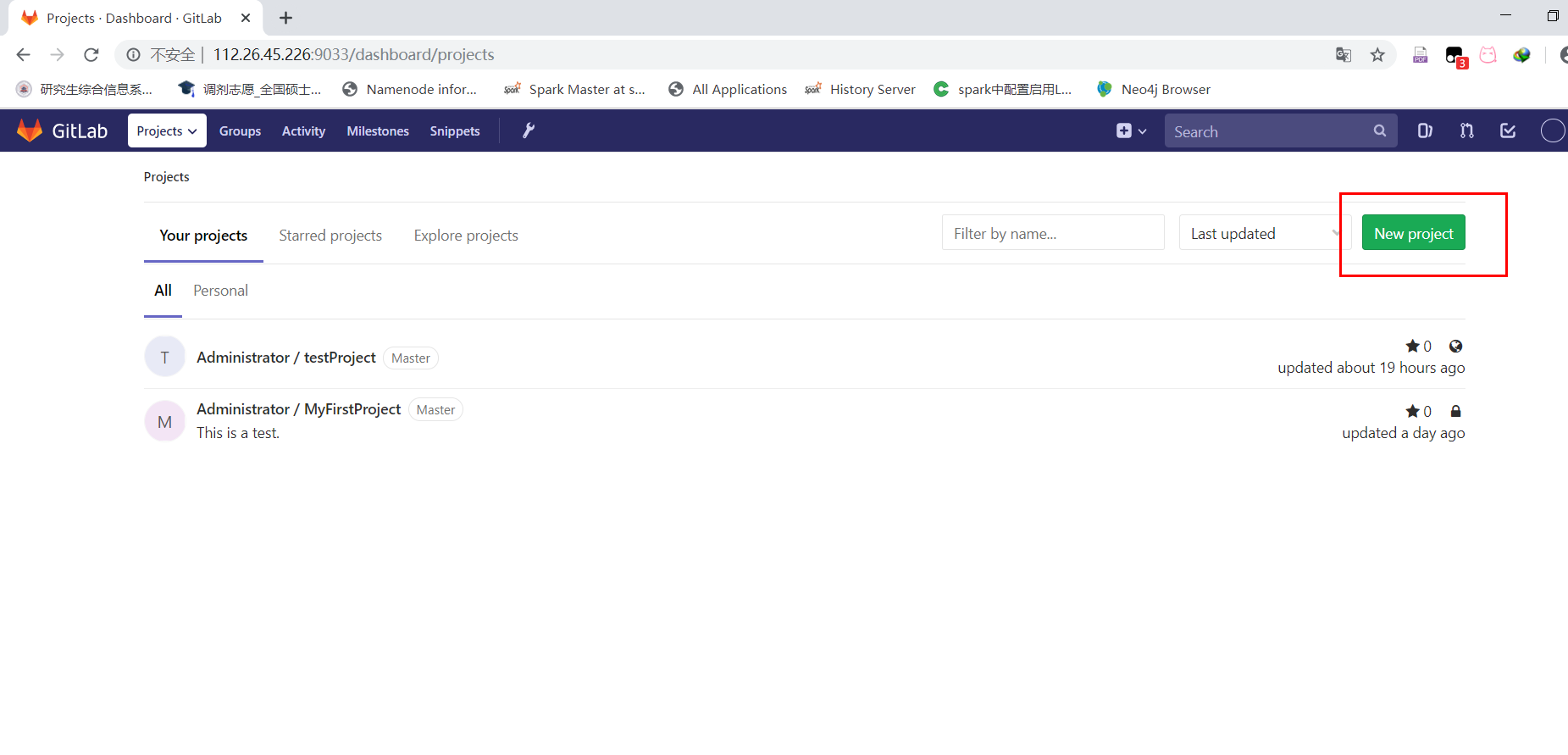
9 点击箭头 发现没有可以推送的远程仓库地址 再点击取消



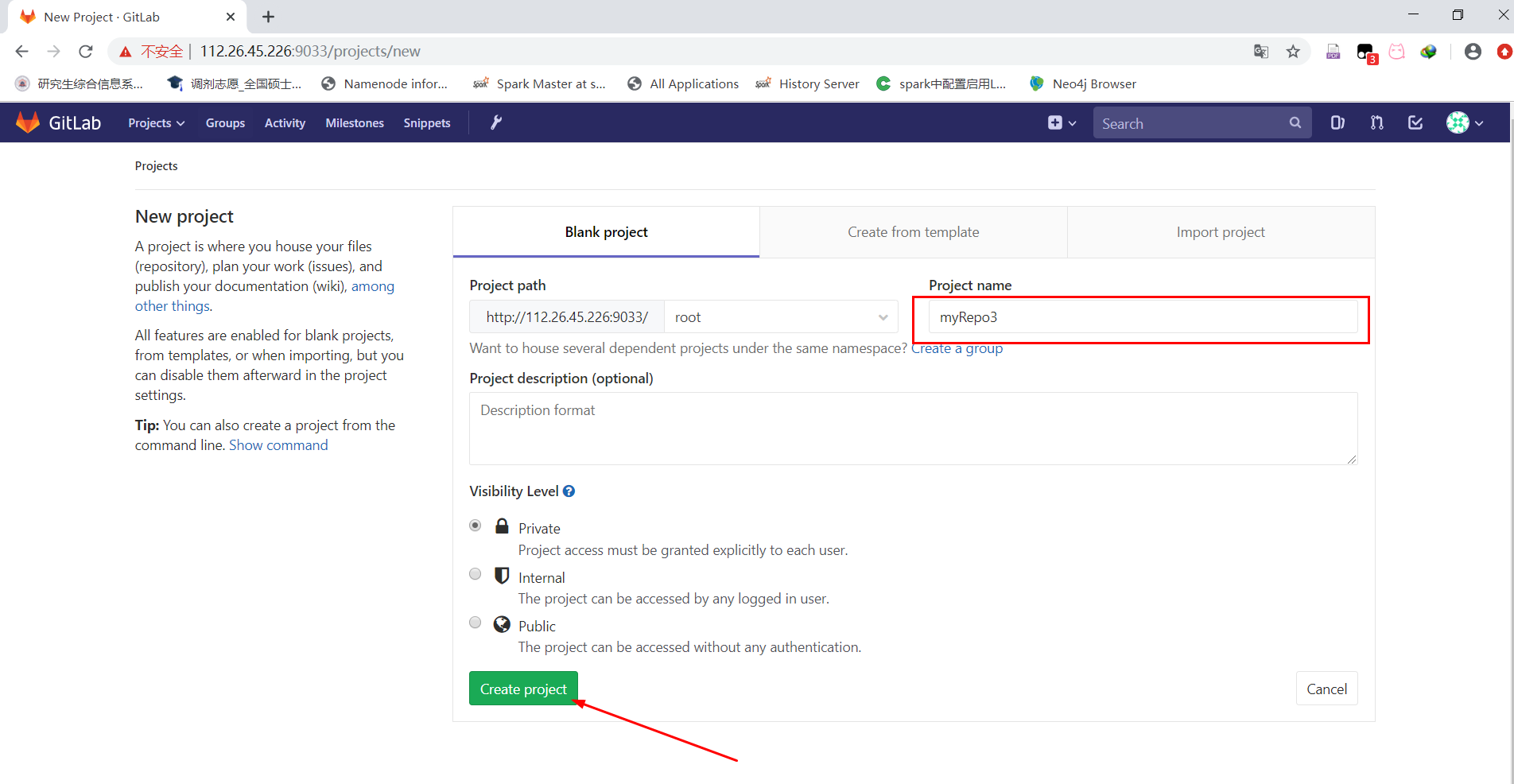
10 到网页点击Projects 准备新建一个仓库



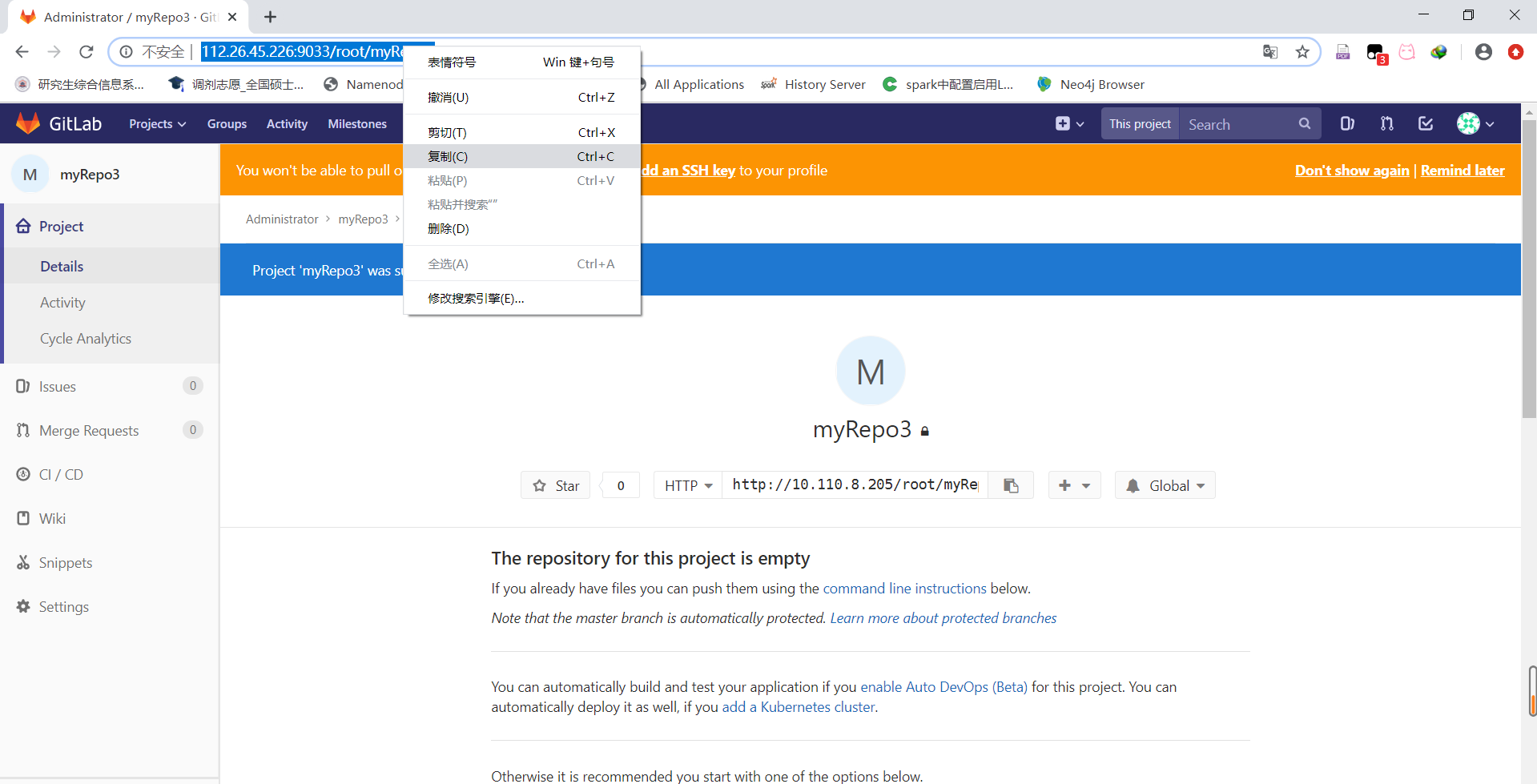
11 点击



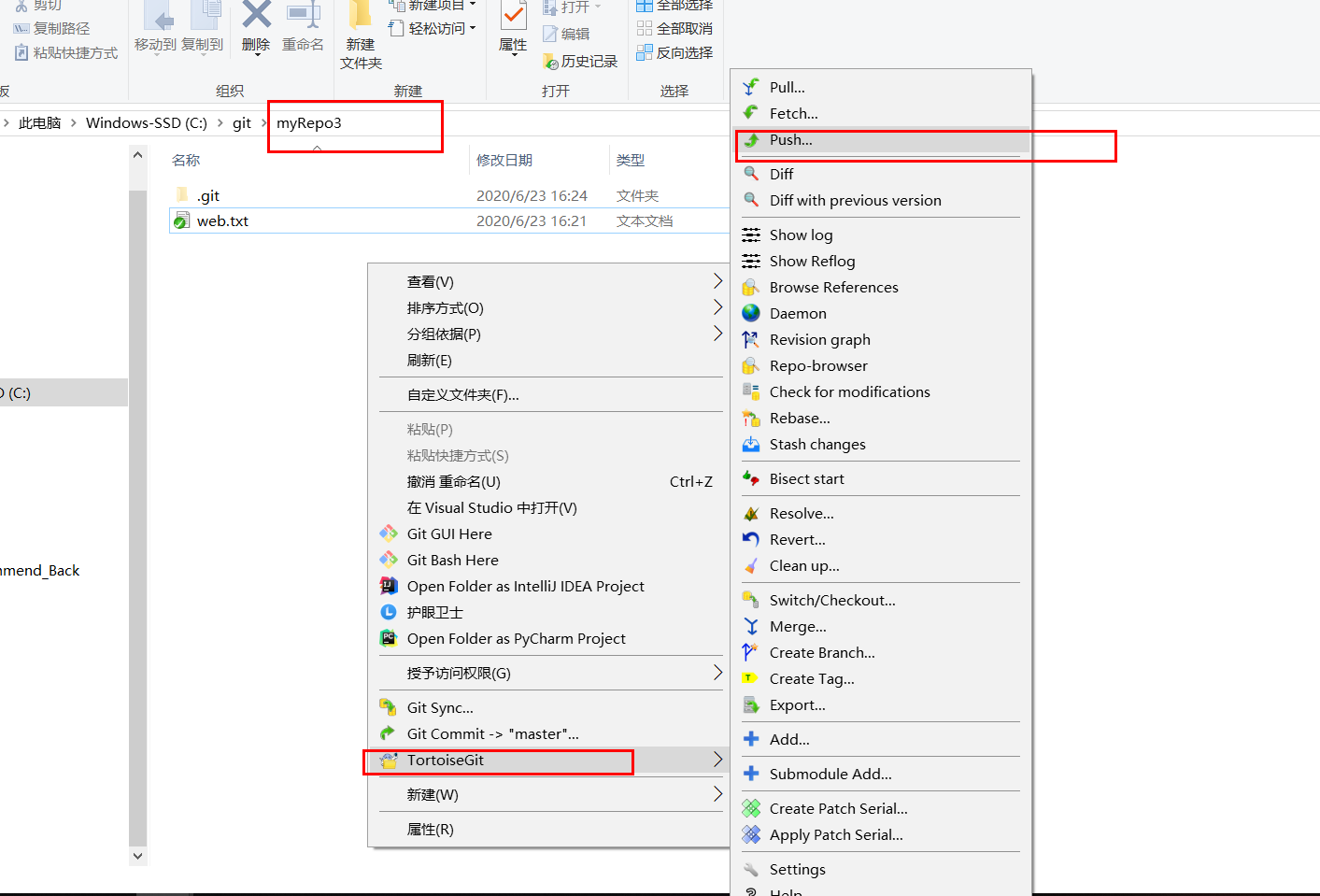
12 在项目名处 写上一个最好与本地仓库一模一样的名字



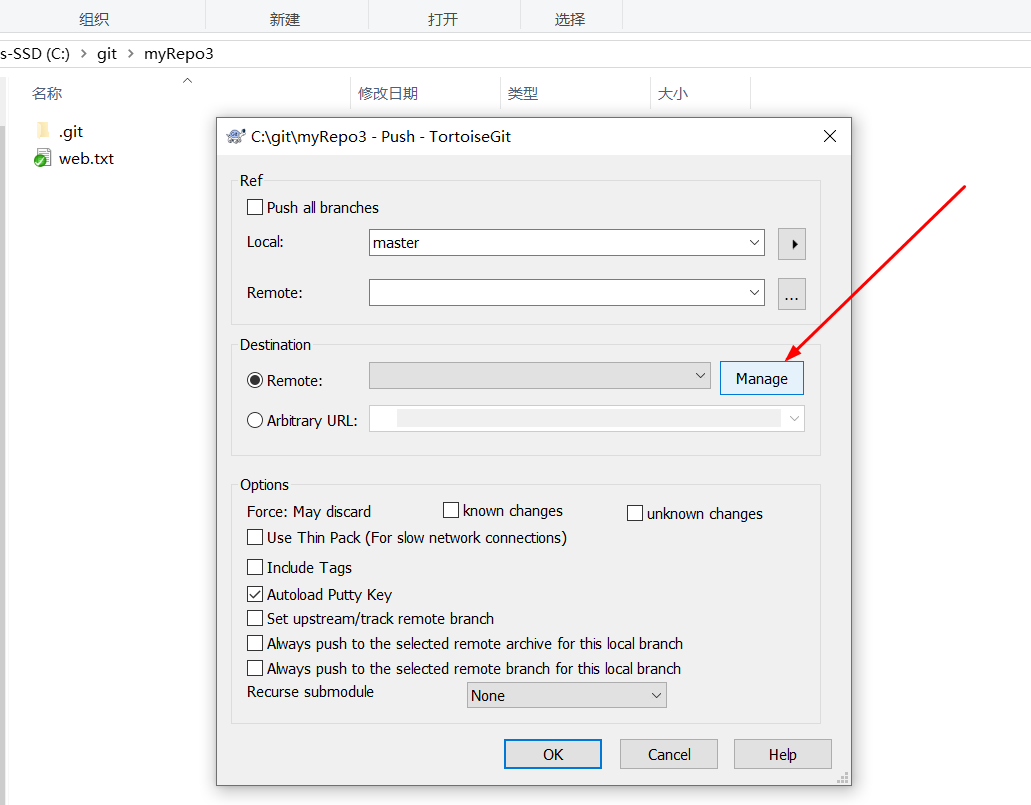
13 复制该远程仓库的网址



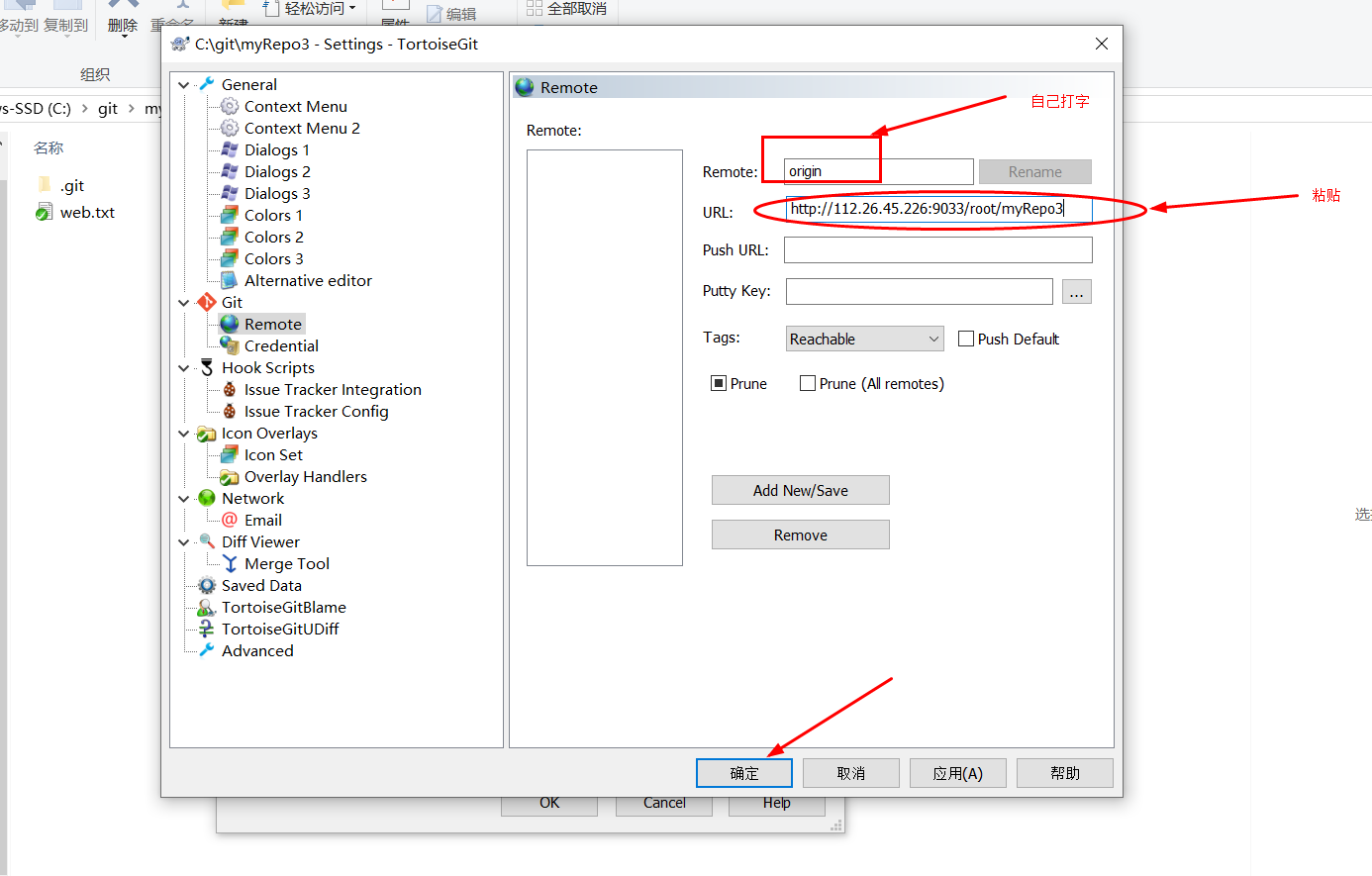
14 回到本地仓库点击Push



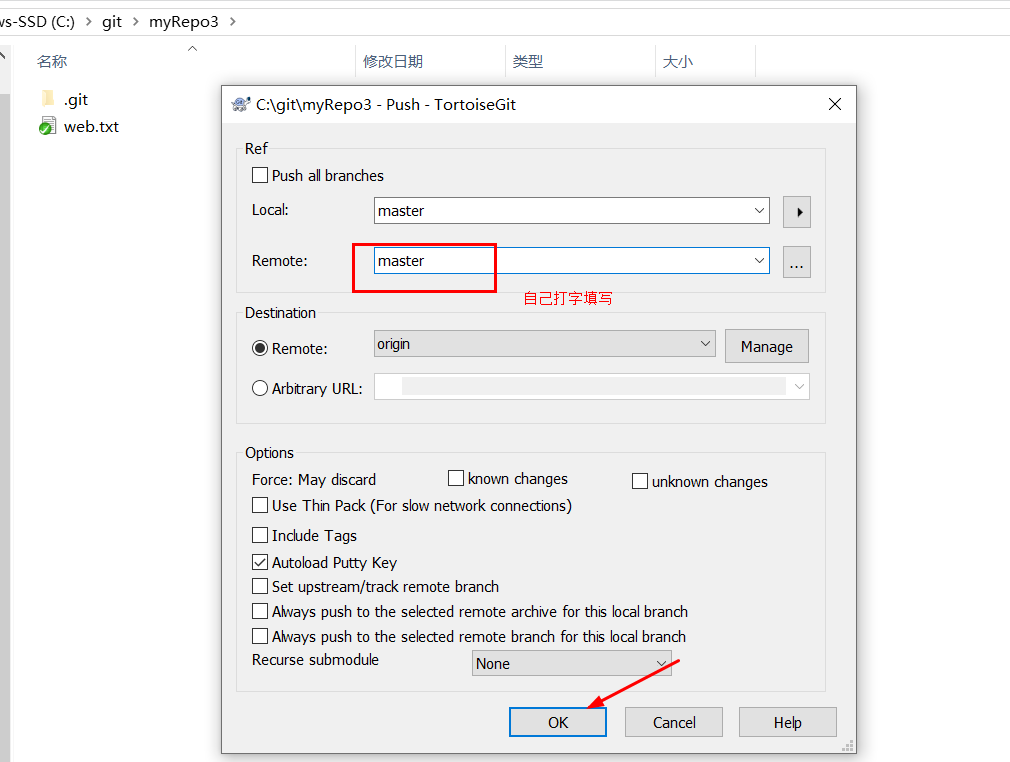
15 点击Manage



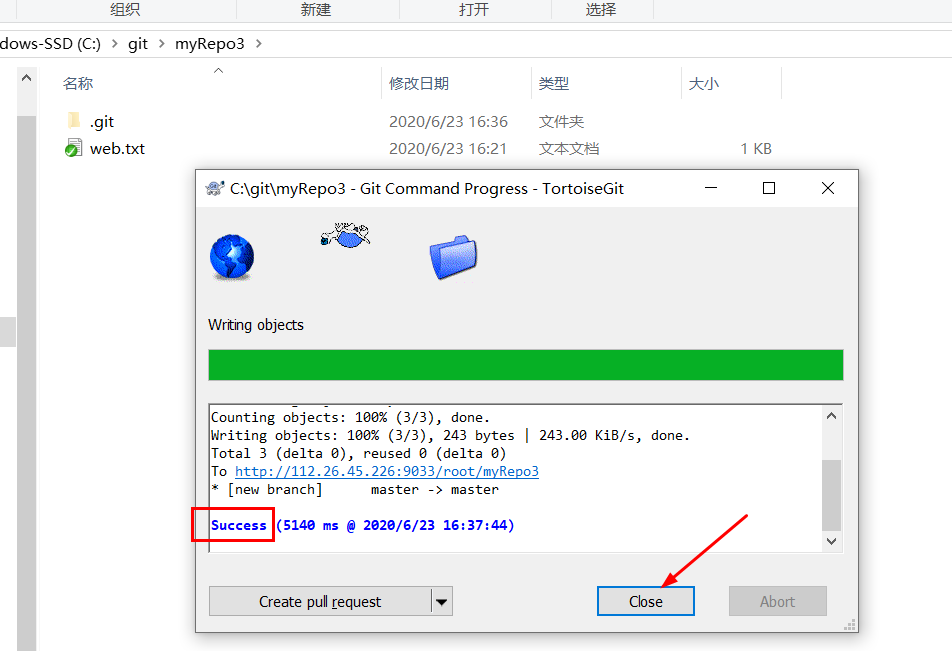
16 填好地址后点击确定



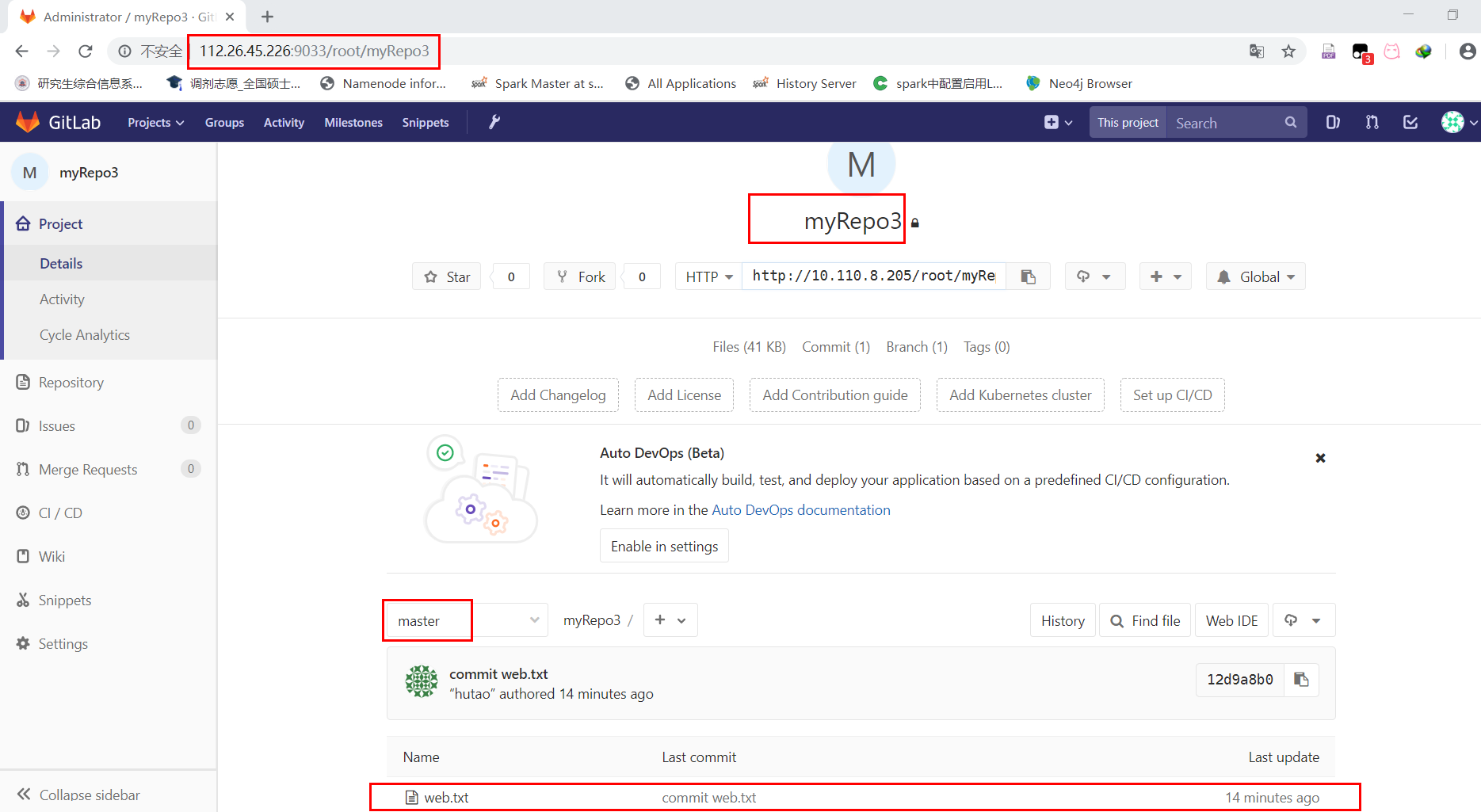
17 填写好自己想要推送的目标分支 后点击确定



18 点击Close

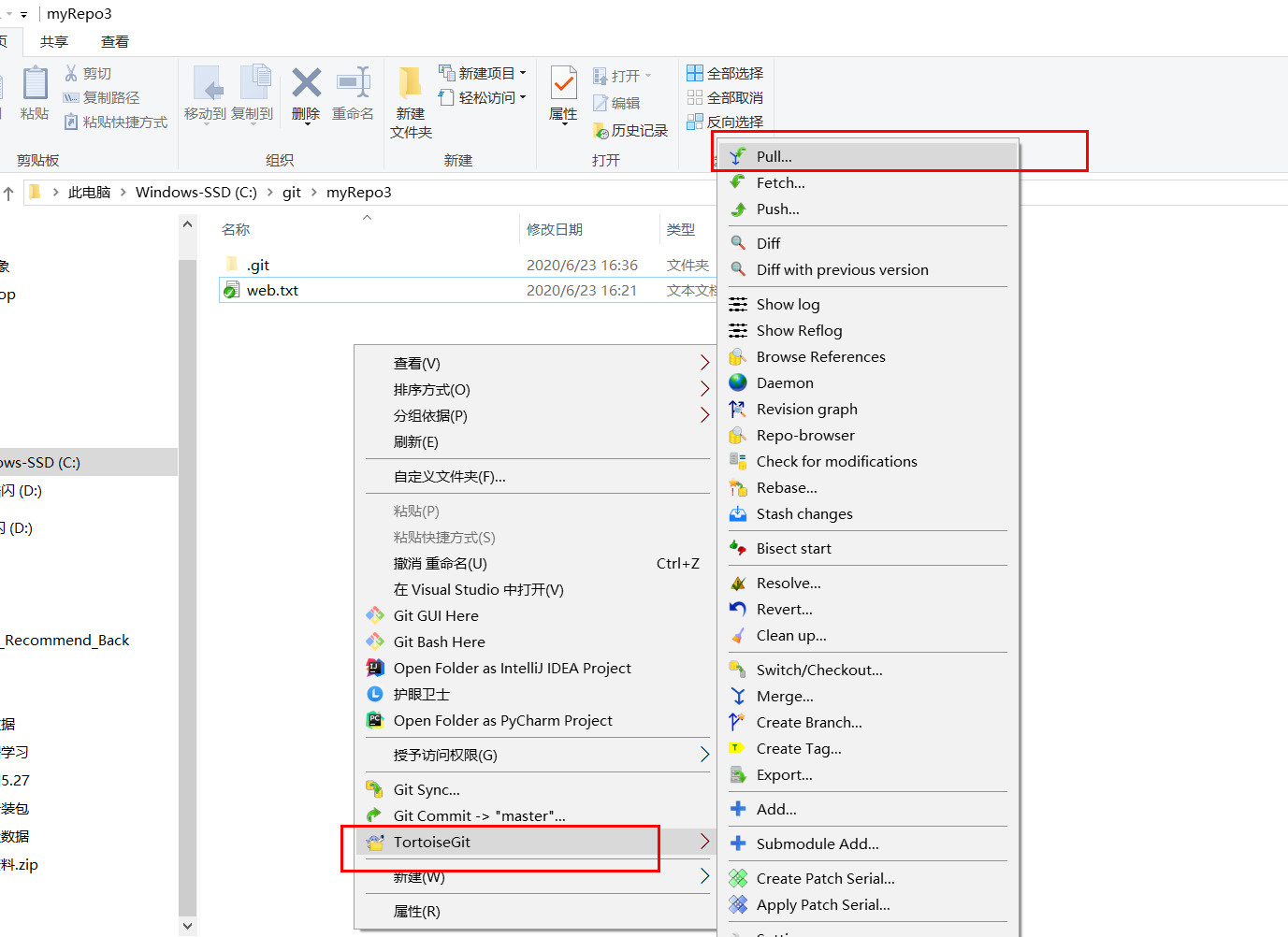


19 到网页刷新界面 出现 web.txt 说明推送成功

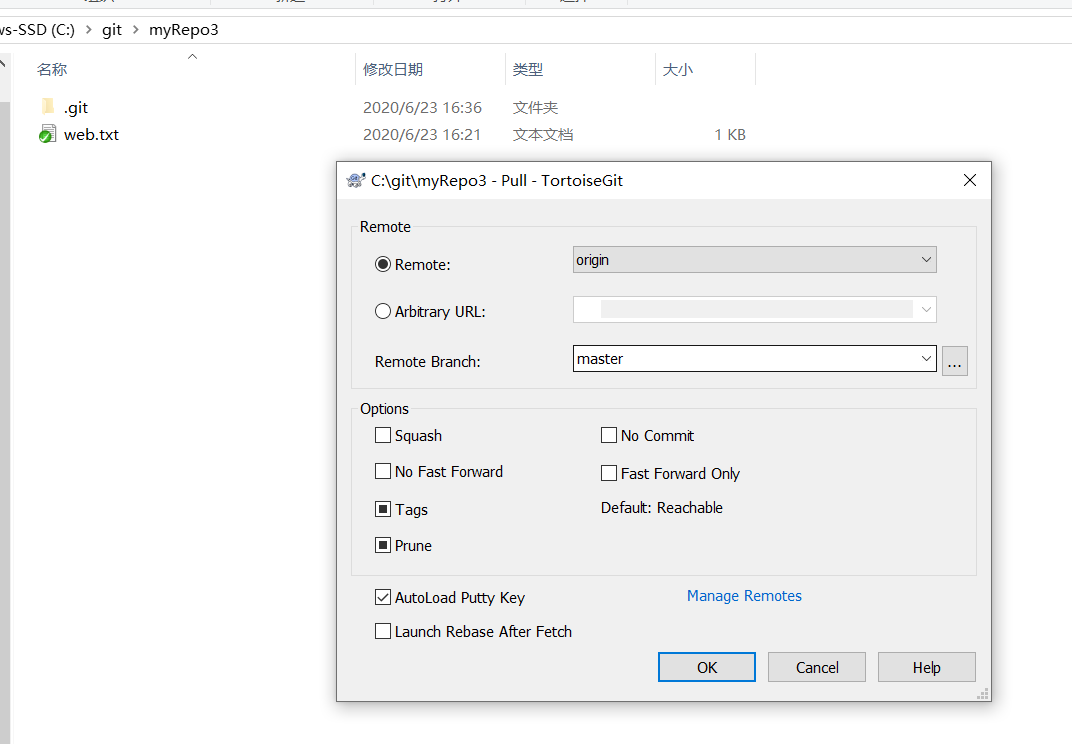


## 拉取远程仓库的代码到本地仓库

1 右键 选择Pull



2 由于刚刚上文已经配置过了 所以相应的参数就会自动补充出来了 点击OK



3 这时 远程仓库若有其余开发人员更新了代码 则自己的本地仓库也会得到更新

