后端项目使用github进行合作开发

提示:因为github是外国网站,所以网速可能比较慢,如果出现上传和下载操作失败,请多试几次

一、github安装与注册

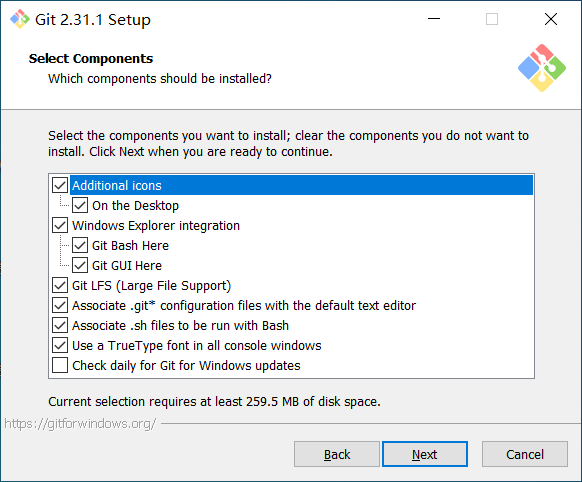
1.下载安装Git

下载地址(windows): <https://gitforwindows.org/>

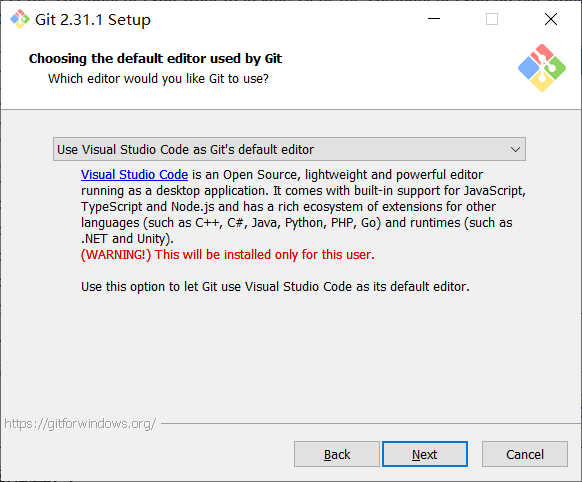
点击”download”下载即可

下载完成后打开安装

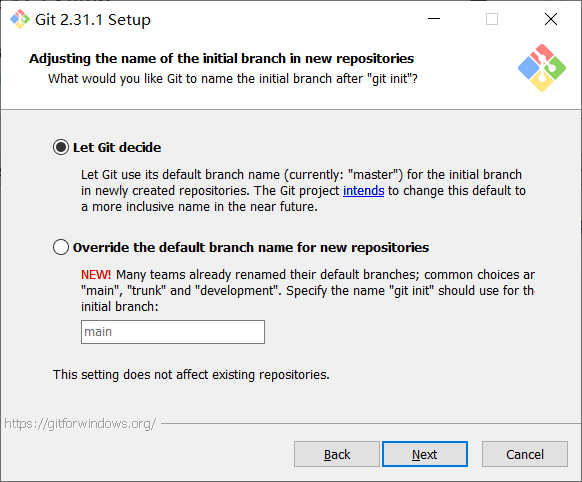
安装配置图



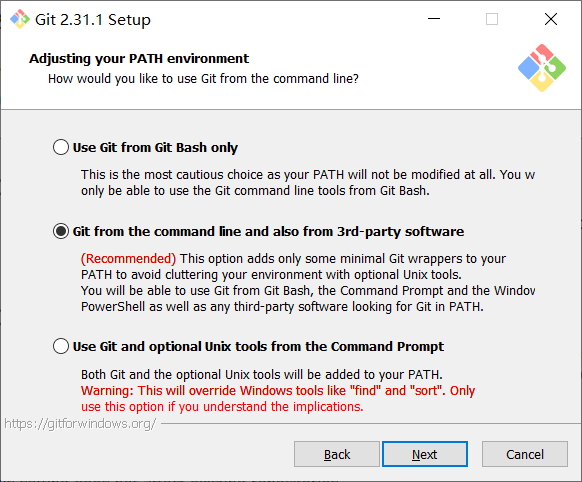
在选择编辑器界面可以选择你喜欢的ide(例如我用的vscode)作为编辑器,如果没有那就直接用vim即可



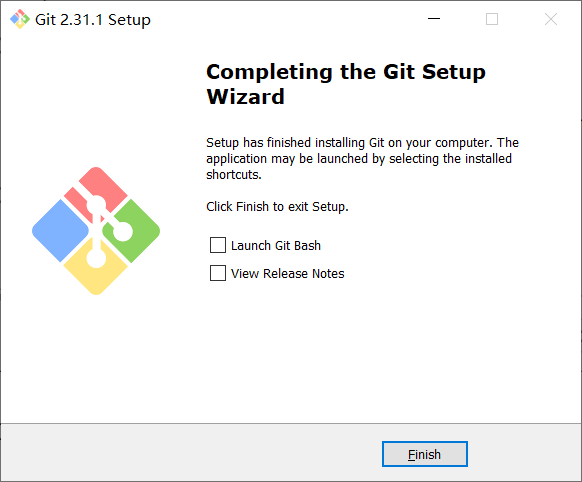
主分支命名直接用默认的master即可



使用方式选第二个(linux可以选第三个)

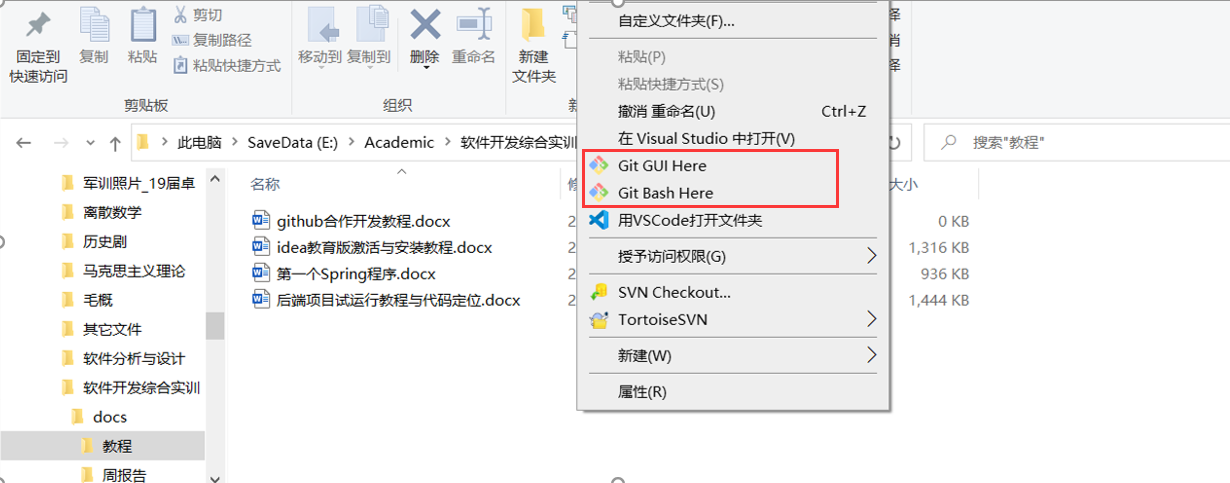


后面的直接一直按next安装即可



安装完成

2.查看安装情况



在任意文件夹右键,看到有两个Git选项即为安装成功

3.注册并登录github帐号(已有帐号可忽略)

进入github.com,自行注册即可

4.将你的帐号名发到群里,邀请你加入仓库

5.任意文件夹右键打开Git Bash

使用一下两个命令绑定你的用户

git config --global user.name “your\_name”

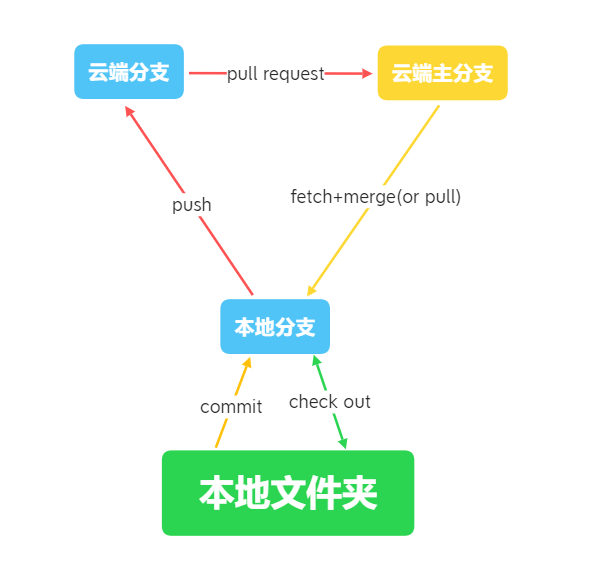
git config --global user.email “your\_email@youremail.com””

6.关闭SSH验证(安全验证比较复杂这里直接关掉)

git config --global http.sslVerify "false"

二、合作开发操作

首先我们需要了解仓库的结构,里面一共有4个空间,我们开始开发前需要将这4个空间创建完成(其中云端主分支已经存在)



(这张结构图十分重要,后面的操作都是参考这个图来构建的,一定要理解)

1. 创建自己的本地仓库并同步云端内容

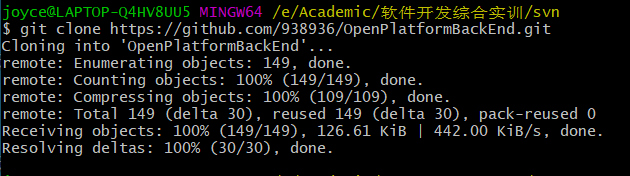
(在成功邀请你加入仓库之后才能操作)

找一个放项目的地方右键Git Bash(项目会新建一个文件夹保存,所以不需要留空这个文件夹)

输入命令

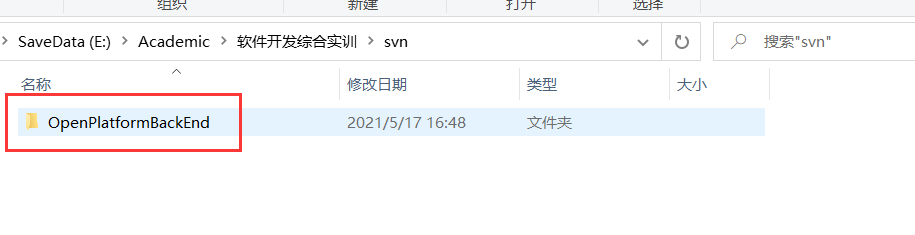
git clone 仓库路径

例如:git clone https://github.com/938936/OpenPlatformBackEnd.git



显示下载了很多文件就是拉取成功了

现在可以在文件夹里面看到对应项目文件夹



此时本地仓库创建完成(位置就是这个生成的文件夹位置)

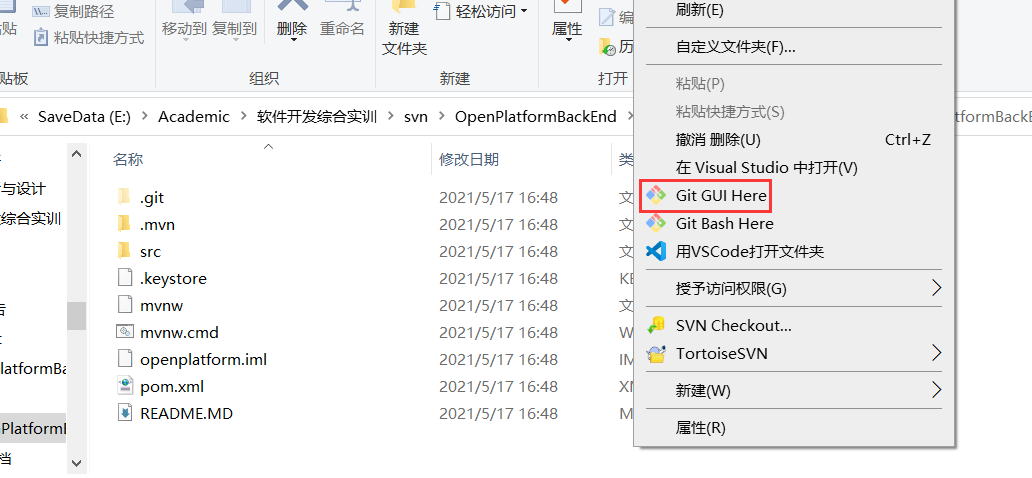
1. 创建自己的分支与云端分支

考虑到安全开发性质,你的所有直接修改应该只能在自己分支里面进行,不应该直接修改master分支(看上面的结构图,所有操作其实都是合并操作而不是覆盖)

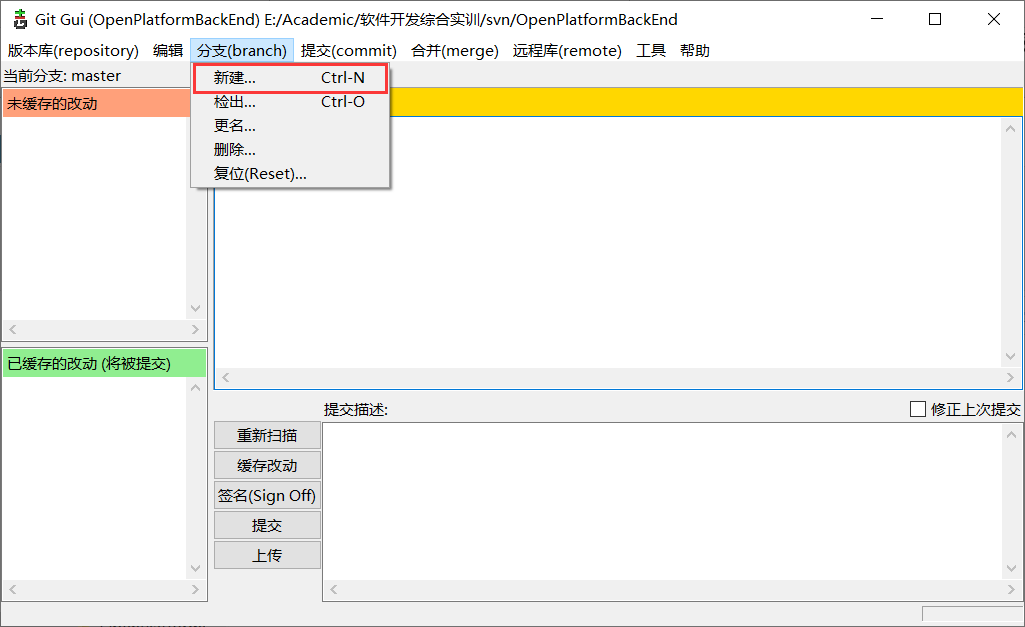
分支其实就是项目的副本,不同分支可以各自做自己的修改,然后合并到master分支上面,保留所有修改的内容,而不是直接全部修改同一个项目导致后面的修改覆盖掉了前面的

现在开始创建自己的分支

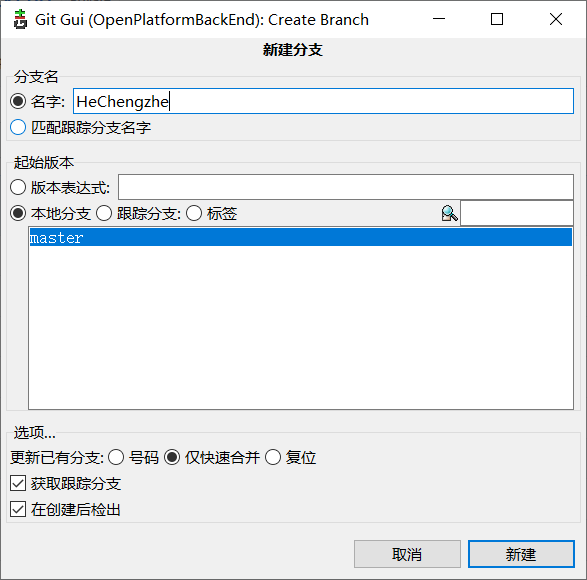
进入项目文件夹,右键Git GUI



点击branch->new

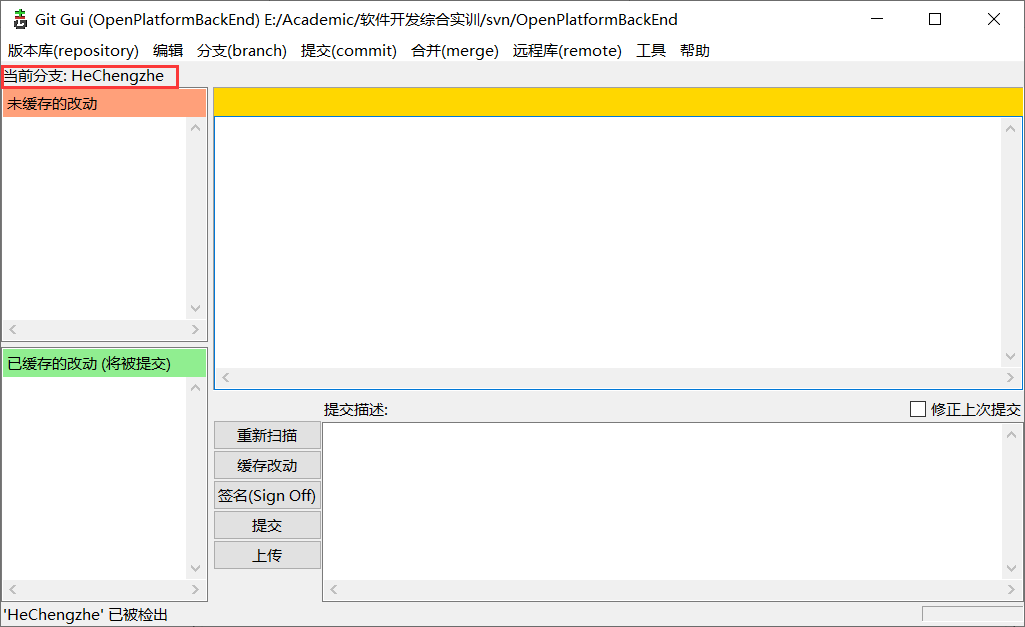


输入自己的名字作为自己分支

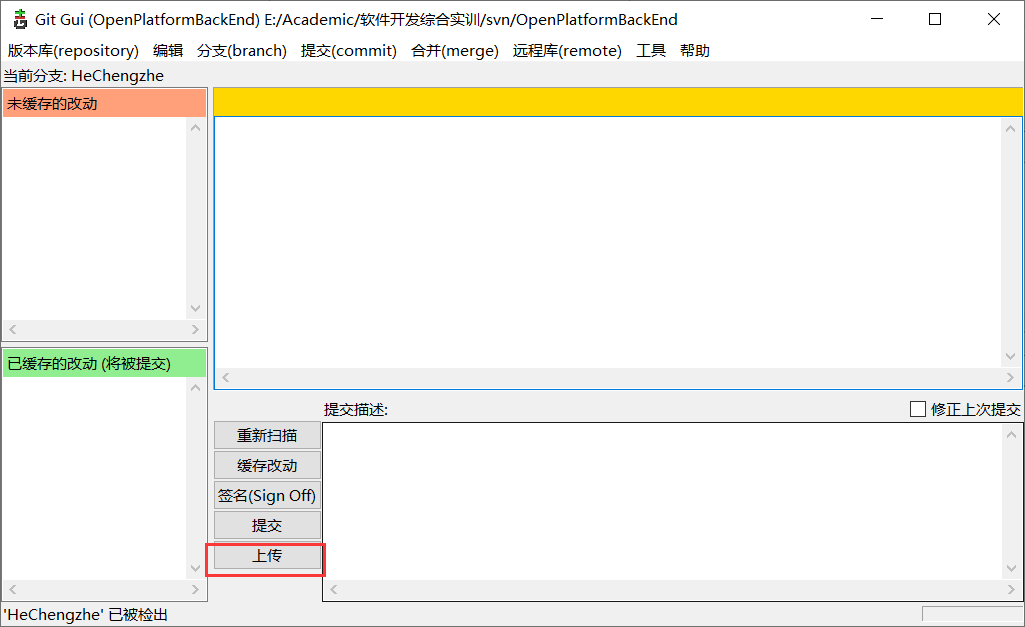


点击新建

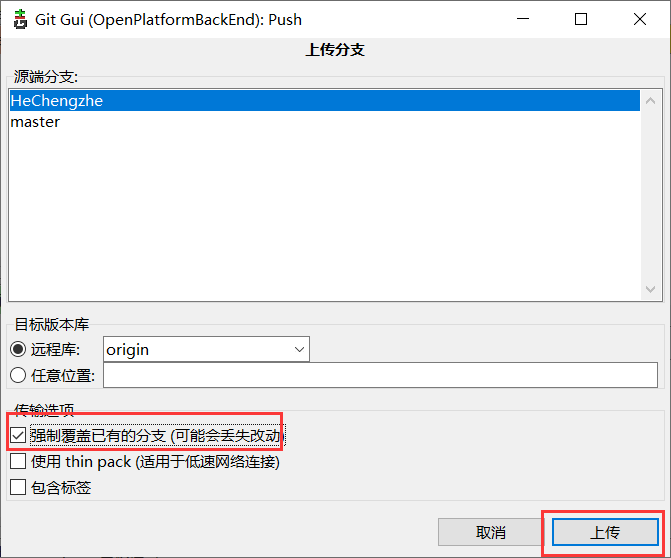
然后查看界面的当前分支是不是自己的



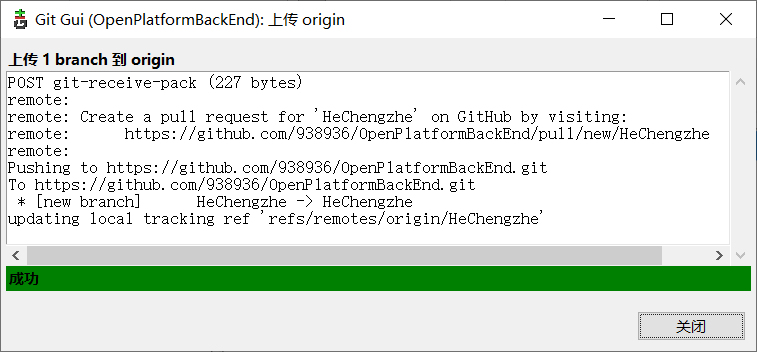
此时本地分支已经创建完毕,需要上传到云端以创建云端分支(见分支结构图)



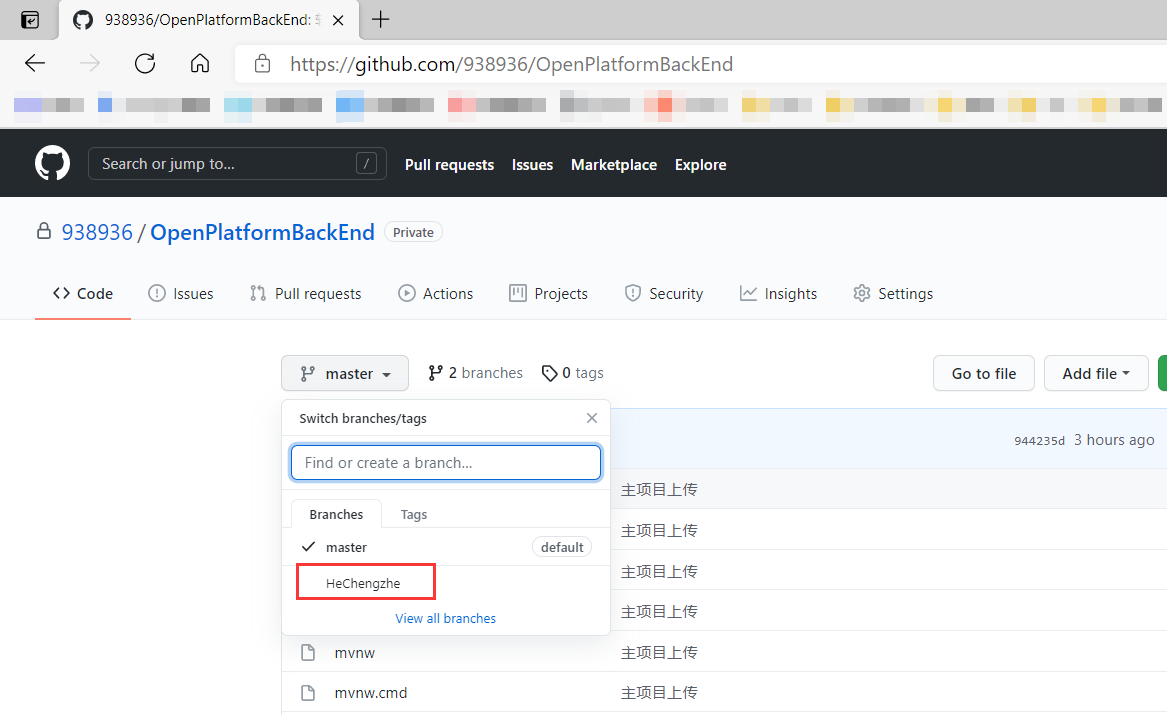
勾选强制覆盖,点击上传



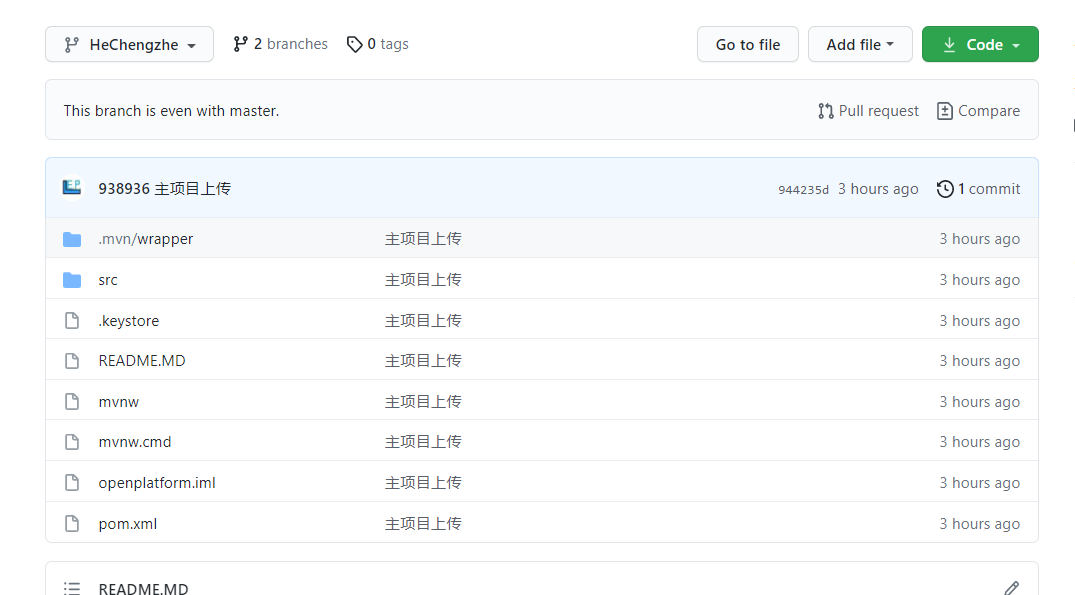
等待上传成功



此时打开浏览器,进入项目云端仓库查看



可以在分支列表中找到自己的分支就是上传成功了,点击进入可以查看里面的内容



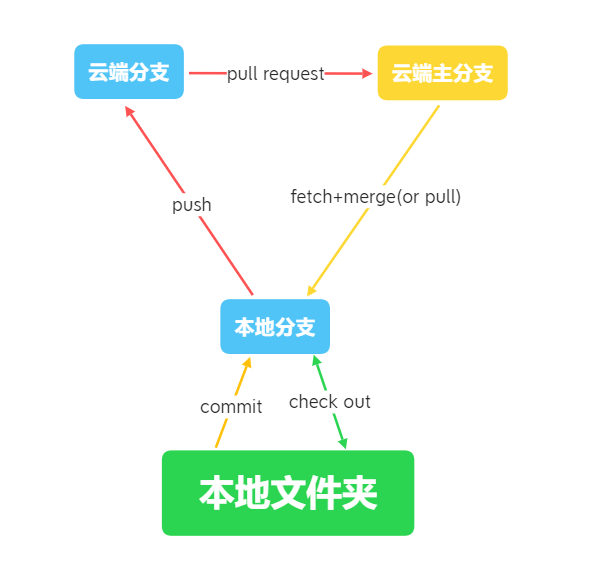
内容应该和master分支一样

此时自己分支也创建完成,分支结构图完整

以后的修改都在自己分支里面进行

接下来了解分支之间的文件推送操作

再次声明分支结构图



1. Check out 检出操作

操作功能:将本地分支的文件同步到本地文件夹,并且将当前操作空间定位到对应的分支

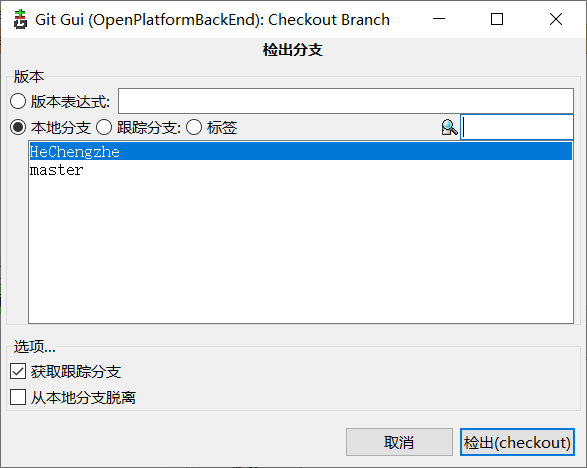
前置条件:本地文件夹不能有未提交的修改(如果有,请进行提交或者进行Reset丢弃修改)

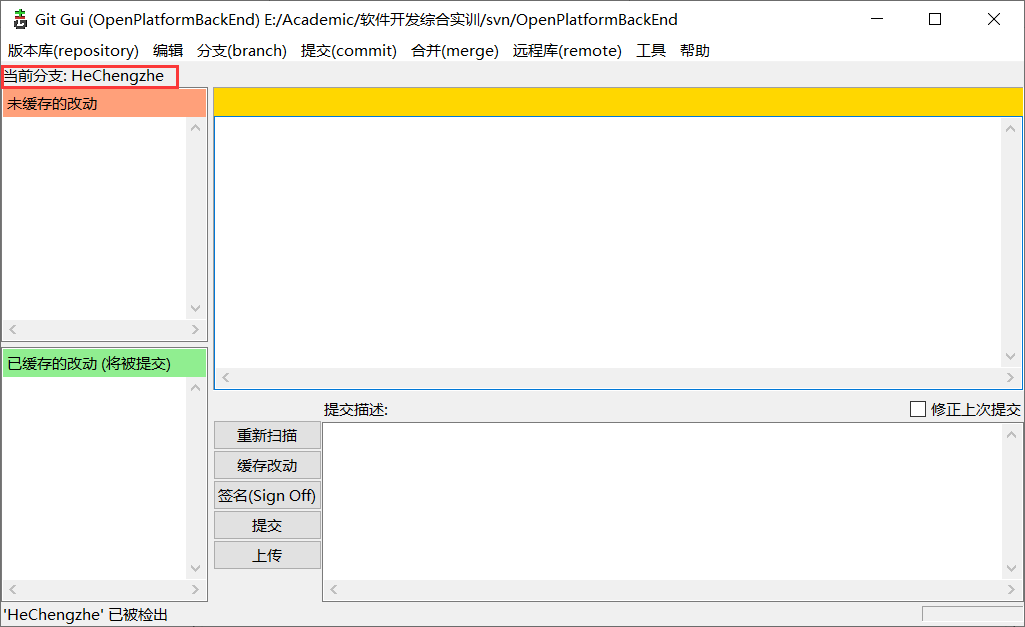
操作方法:

Git GUI->branch->check out->选择检出分支

检出时勾选”获取跟踪分支”就会同时从云端分支获取数据,不勾选则直接从本地分支检出文件

后面经过实践,只需要检出一次自己的分支即可,本地不需要检出master分支,也不能直接修改master分支





1. commit 提交操作

操作功能:将本地文件夹的修改提交到本地分支

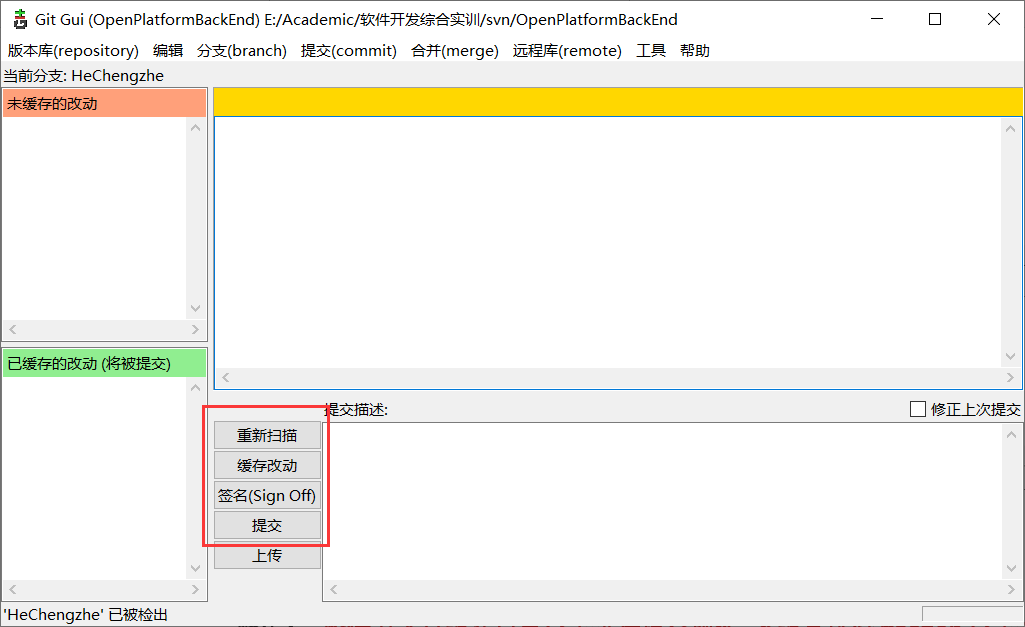
前置条件:已检出到对应分支

警告:一般情况下只能在自己分支进行提交操作,不能直接对master分支进行提交

操作方法:

GUI->[先检出到自己分支]->重新扫描->缓存改动->填写提交描述(填写你这次提交修改了什么)->点击提交

因为非公开库没有安全问题,所以不需要签名



1. merge合并操作

操作功能:将一个分支的修改合并到另一个分支,合并操作会保留两个分支的修改,而不是单纯的覆盖,也是合作开发的根本目的(避免同步开发导致的覆盖)

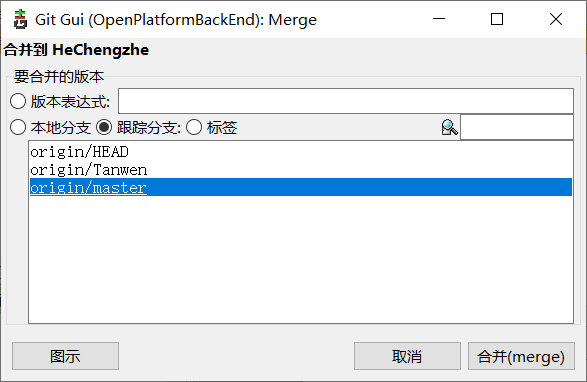
前置条件:合并目标分支需要没有未提交的修改,并且版本不能严格高于源分支。

警告:合并目标一般情况下只能是自己分支,merge操作用于将云端master的最新修改同步到自己的本地分支

操作方法:

GUI->[检出到自己分支]->merge->本地分支->选择源分支->合并

跟踪分支就是指云端分支



1. push推送操作

操作功能:将本地分支的修改上传到云端

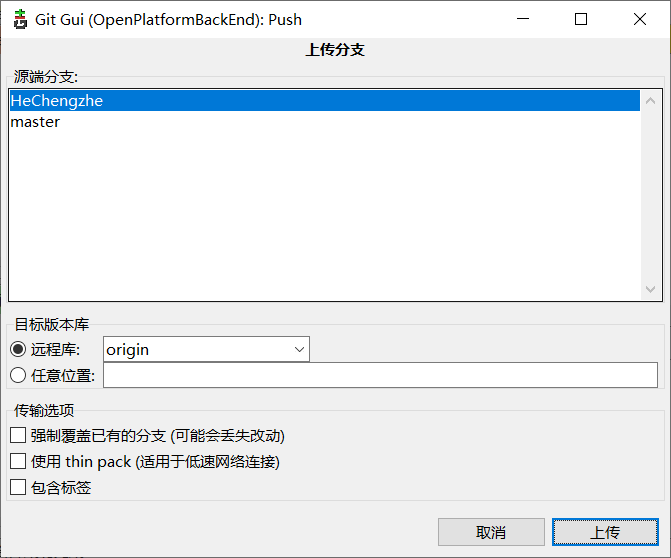
前置条件:已经将修改提交到本地分支

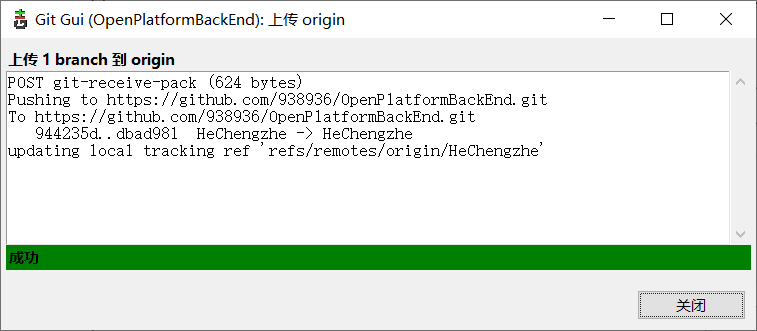
警告:上传操作只能在自己分支进行,不能直接上传到master分支

操作方法:

GUI->push->选择自己分支->上传

如果经常失败可以勾选使用thin pack提高稳定性





1. fetch 接收操作

操作功能:将云端分支的修改下载到本地分支,通常用于获取master分支的最新修改,注意此时只是下载了修改,但没有同步到文件夹,需要merge操作来同步

前置条件:本地分支中不能有未提交的修改,本地分支版本不能严格高于云端分支

操作方法:

方法1(推荐):打开Git Bash->输入命令 git fetch ––all

方法2:打开GUI->remote->fetch->origin

注意这个操作可能失败,对网络要求较高,多试几次即可

1. pull request 拉取请求操作

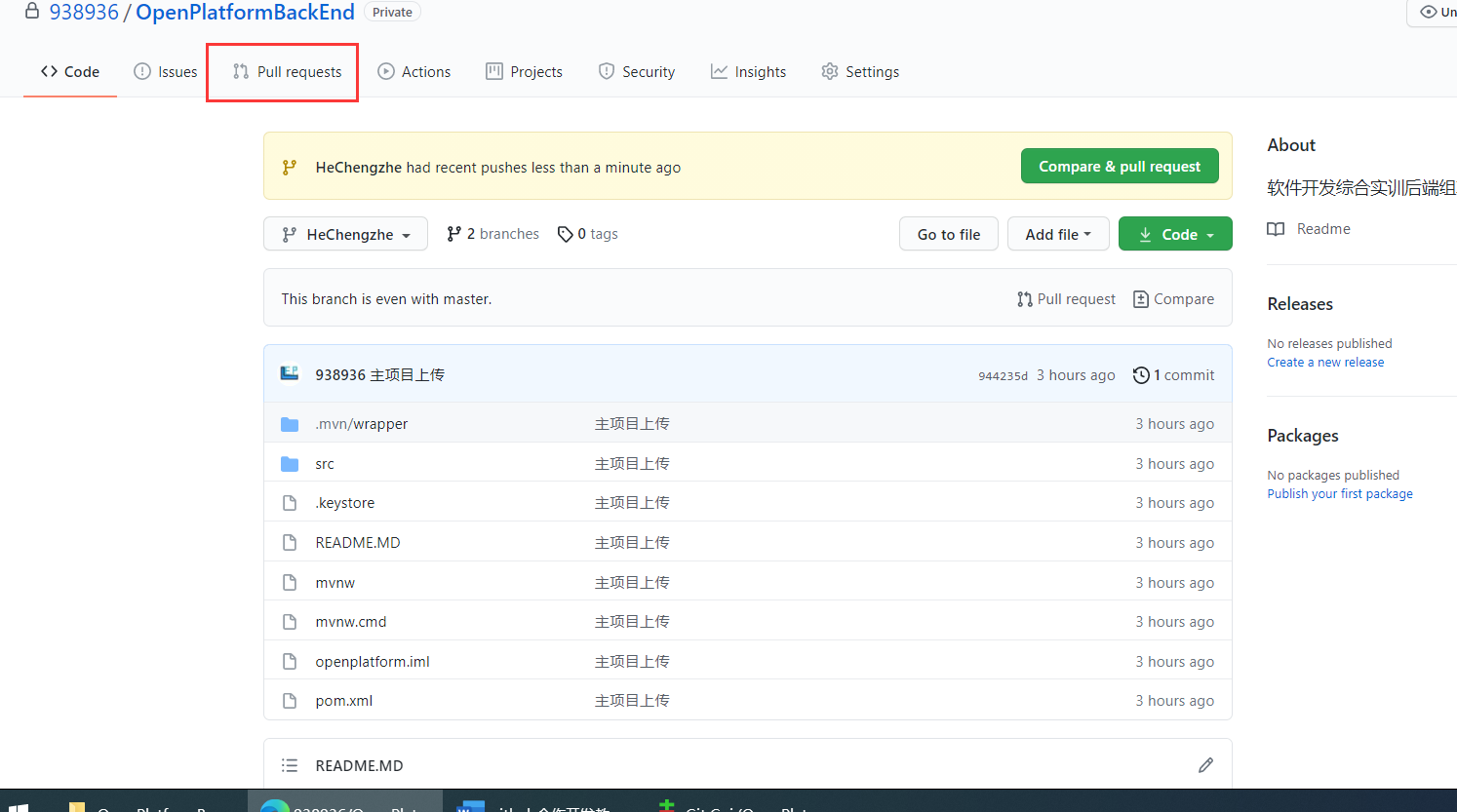
操作功能:将一个云端分支的修改合并到另一个云端分支,这个操作将作为修改云端master分支的唯一方式

前置条件:没有合并冲突(有就需要解决),目标分支的版本不能严格大于源分支

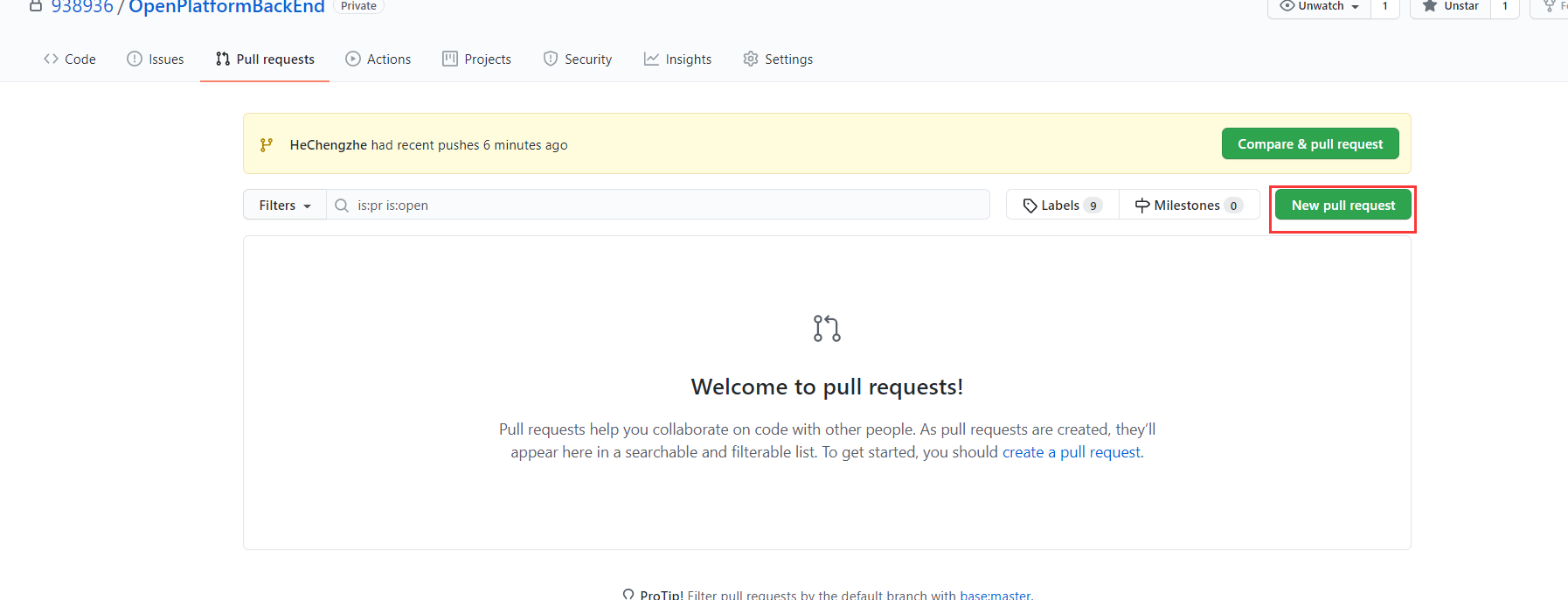
操作方法:

PR操作将全部在云端分支进行

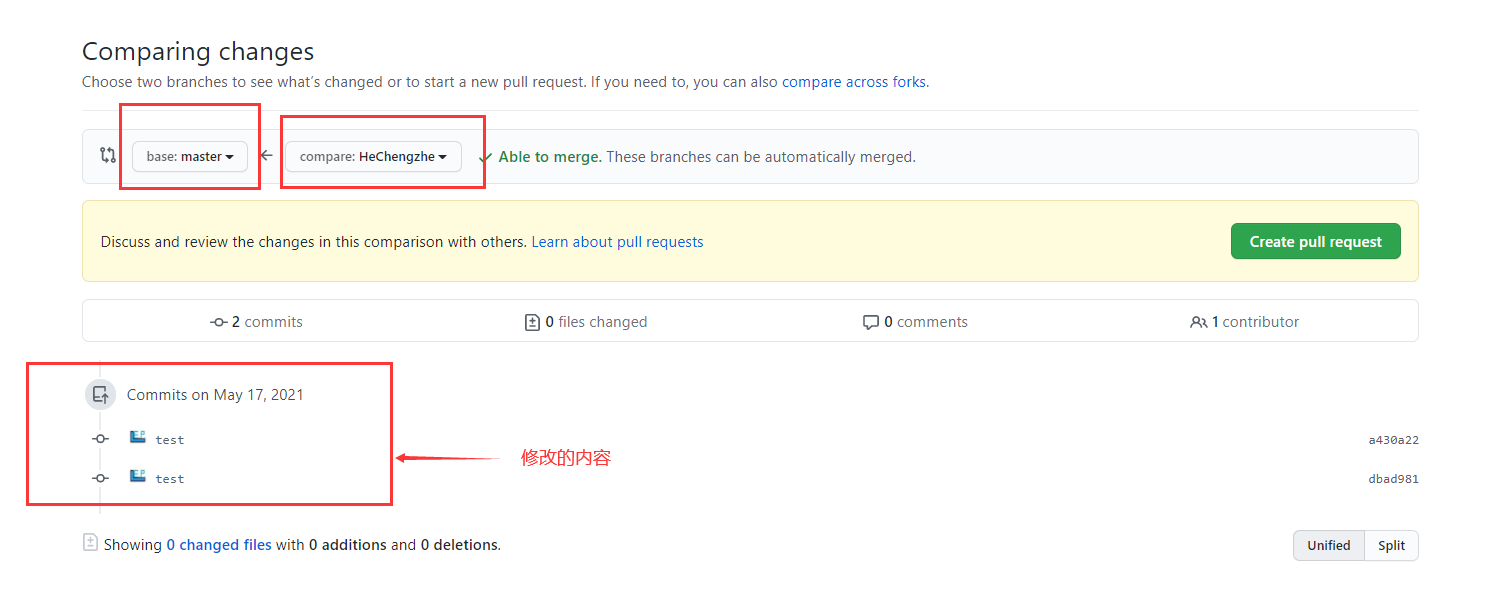
打开浏览器,打开仓库,选择pull request



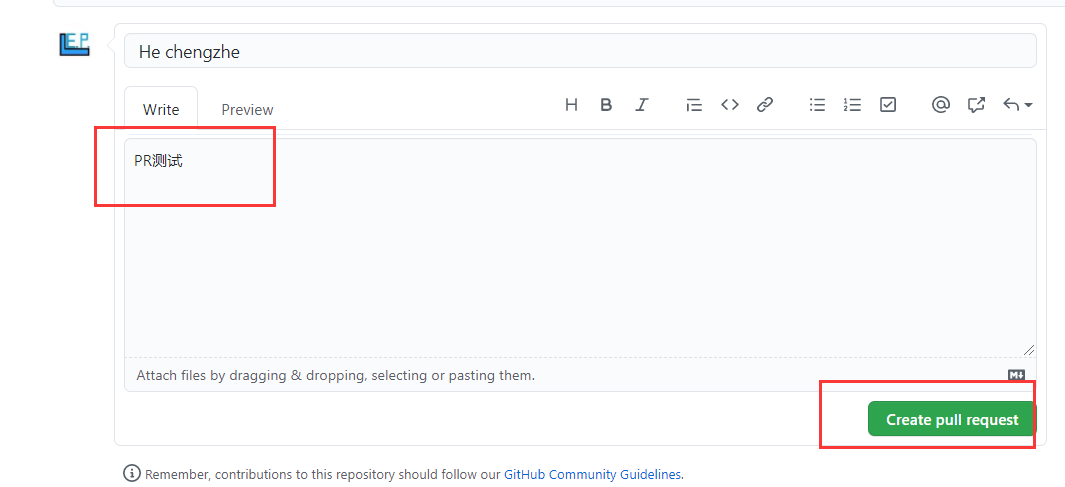
点击新建PR(或者刚更新了内容上面有黄色的提示可以直接点)



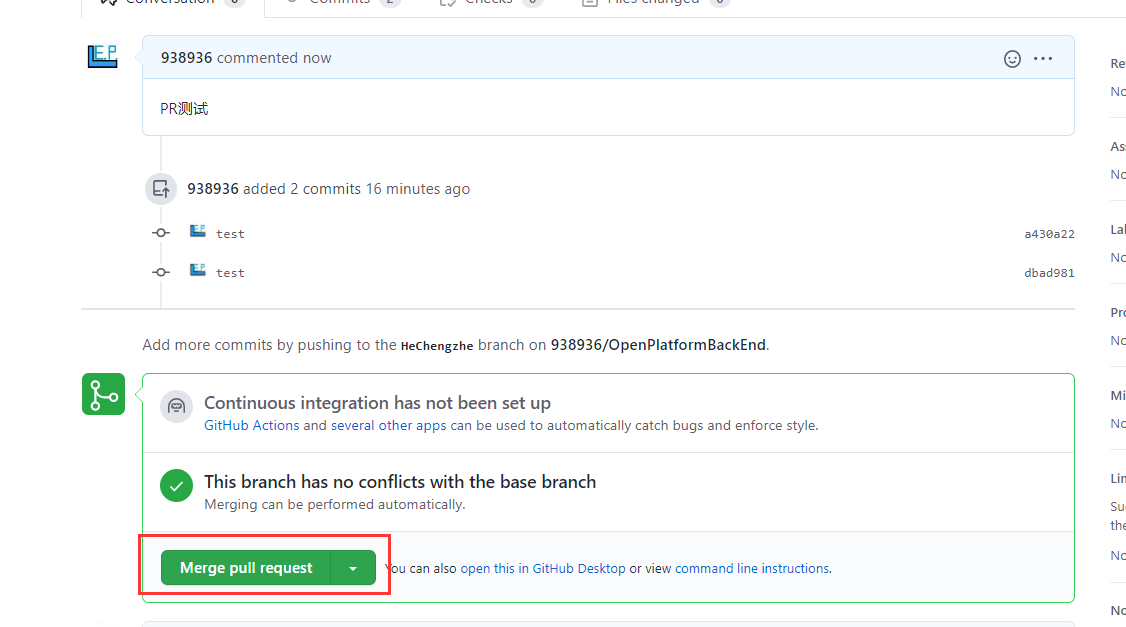
选择自己分支合并到master



点击Create PR,填写修改的详细内容(必要,告诉其它人你干了什么),选择创建

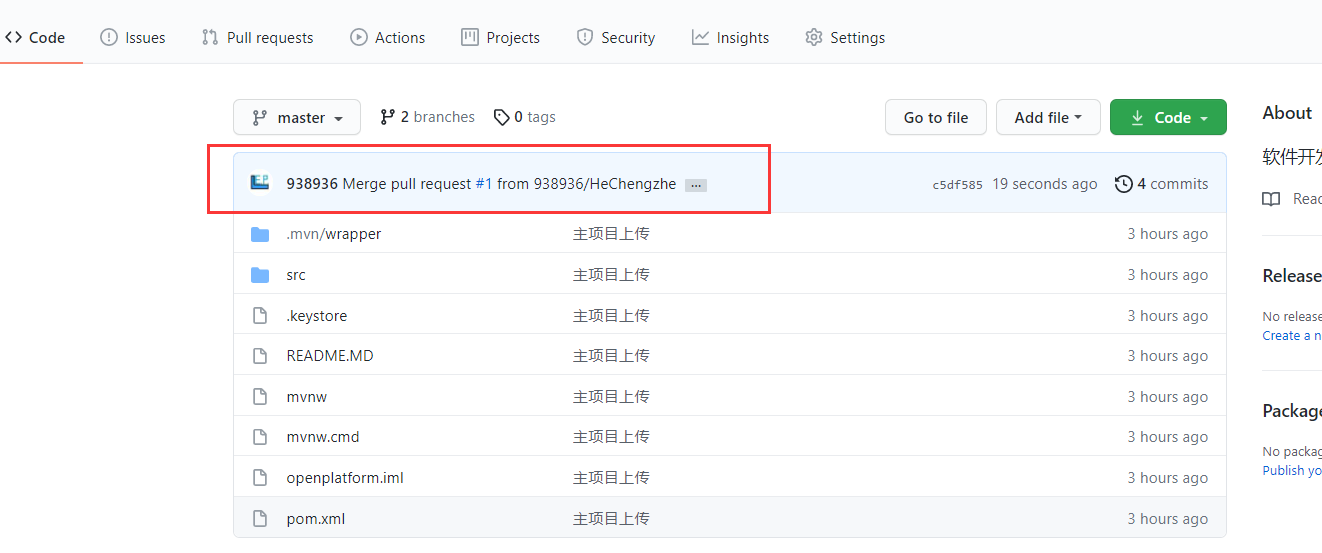


系统检测到没有冲突,可以点击合并



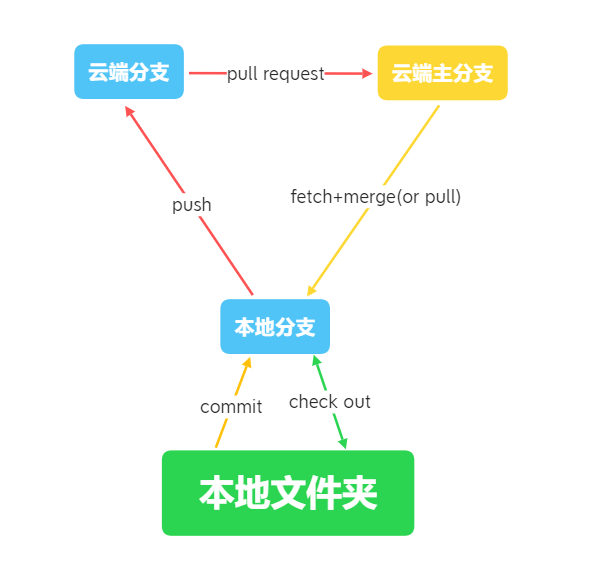
再点击Confirm merge确定合并即可

此时你的分支的修改已经成功更新到云端的master,其它人可以在自己的git上面fetch获取这个更新,并且同步到他自己分支进行开发



至此,所有合作开发操作已经介绍完毕,可以进行开发,接下来是一些开发流程

再次请出分支结构图



1. 在你准备进行开发前,应该先将最新的master下载到本地,并且同步到自己分支

操作流程:

1. 检出本地分支
2. 将云端master分支内容fetch+merge到本地分支
3. 将本地分支的内容push到自己的云端分支
4. 在你开发完成后,将开发提交(commit)到本地分支,然后准备提交到master

操作流程:

1. commit本地分支,填写简要修改内容
2. 将本地分支的修改push到云端分支
3. 进入网页进行PR,合并到master分支,并填写修改内容
4. 在你完成了提交后,你应该再次重复1的操作,将云端的最新内容再同步到本地,并打开你的ide运行项目测试是否能运行,如果不能运行则需要回退PR,修改错误后再上传(如果提交之前能运行而合并之后不行,那肯定是出现了冲突,需要解决冲突)

合作开发教程结束,关于冲突处理等其它问题在遇到冲突后可以自己视情况解决