



序号:

廣東工業大學

课 程 实 验

Git 软件的使用

课程名称 软件系统设计与体系结构

专业班级 17 软件工程 2 班

学 号 3117007142

学生姓名 陈悦演

指导教师 郝彦军

2020 年 7 月

目 录

一、实验目的.....	1
二、实验平台.....	1
三、实验前准备.....	1
四、实验内容.....	1
4.1 Git 安装流程.....	1
4.2 创建本地仓库.....	6
4.3 Git 的提交.....	7
4.3 创建分支.....	10
4.4 Git 冲突.....	11
五、实验总结.....	14

一、实验目的

1. 学习 Git 软件的安装
2. 学习 Git 的简单使用，增加文件，提交，创建分支，增加远程，下拉、上推，合并
3. 学习 Git 的冲突处理

二、实验平台

实验平台选择 Windows 10 系统，该系统使用人群广泛，交互界面十分良好，而且操作起来简单方便。

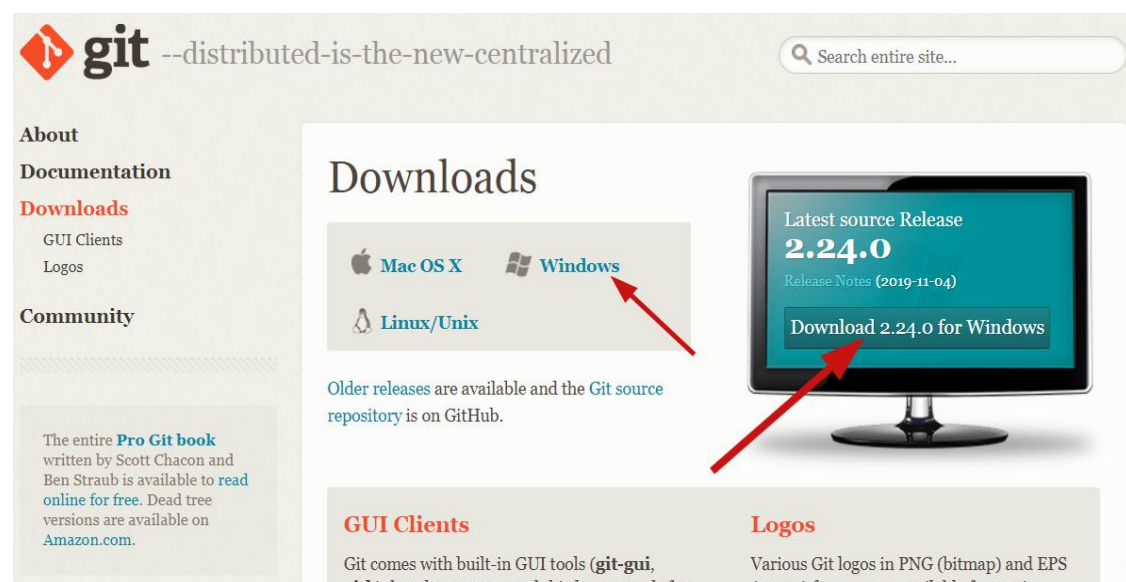
三、实验前准备

1. 搜索浏览 [Git 官网](#)，根据自己所使用的操作系统下载相应的 Git 操作工具。
2. 翻阅相关资料，掌握分布式版本控制系统 Git 的基础知识。
3. 查找 Git 的使用教程，熟悉 Git 命令以及 Git GUI 的使用，以及一些更高级的操作。

四、实验内容

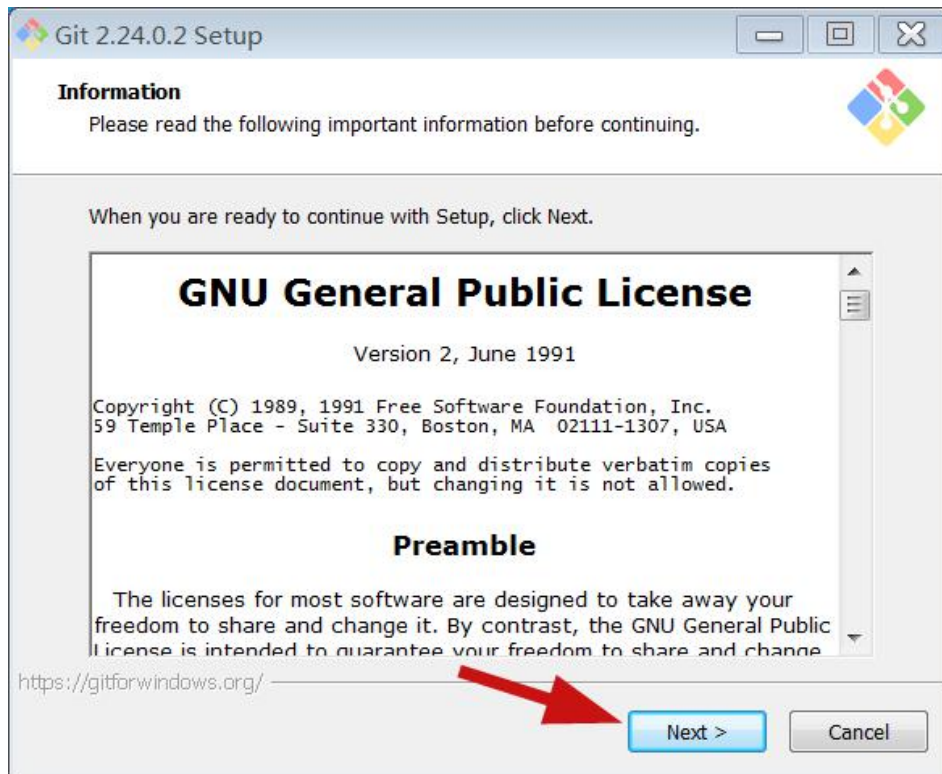
4.1 Git 安装流程

(1) 获取 Git 安装程序



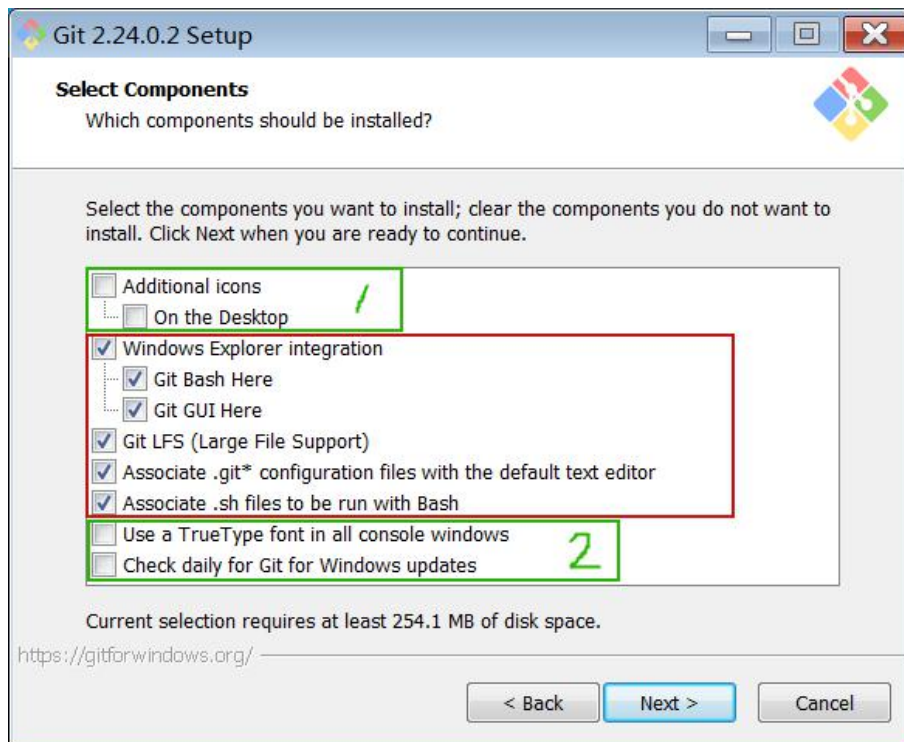
到 Git 官网下载，根据自己的操作系统进行下载，这里采用 window 浏览，Git 官网自动之别到了我使用的操作系统，点击下载后可以得到 Git-2.24.0-64-bit.exe 文件。点击进入安装过程

(2) 使用许可声明



阅读完并点击 next 进行下一步安装。

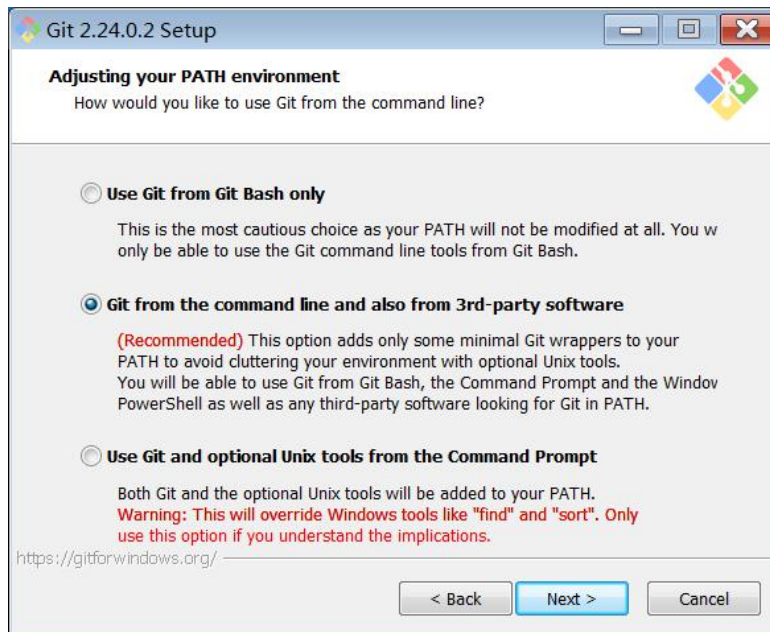
(3) 选择安装的组件



我们可以选择：

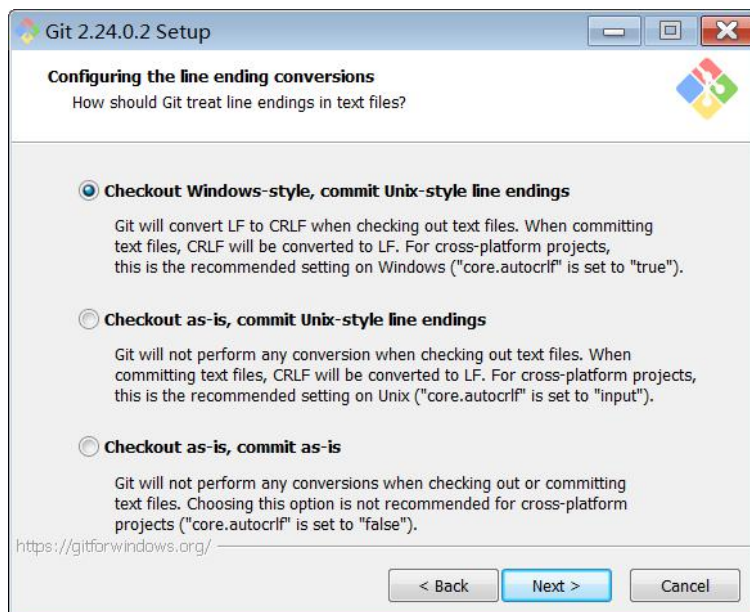
1. 绿框表示是否创建快捷键图标和 window 更新相关，建议不要进行勾选
2. 在右键菜单中添加"Git Bash Here"和"Git GUI Here"，这样我们可以在任意文件路径中通过右键菜单打开 Git
3. 关联 .git 和 .sh 文件到 Git

(4) 设置路径环境变量

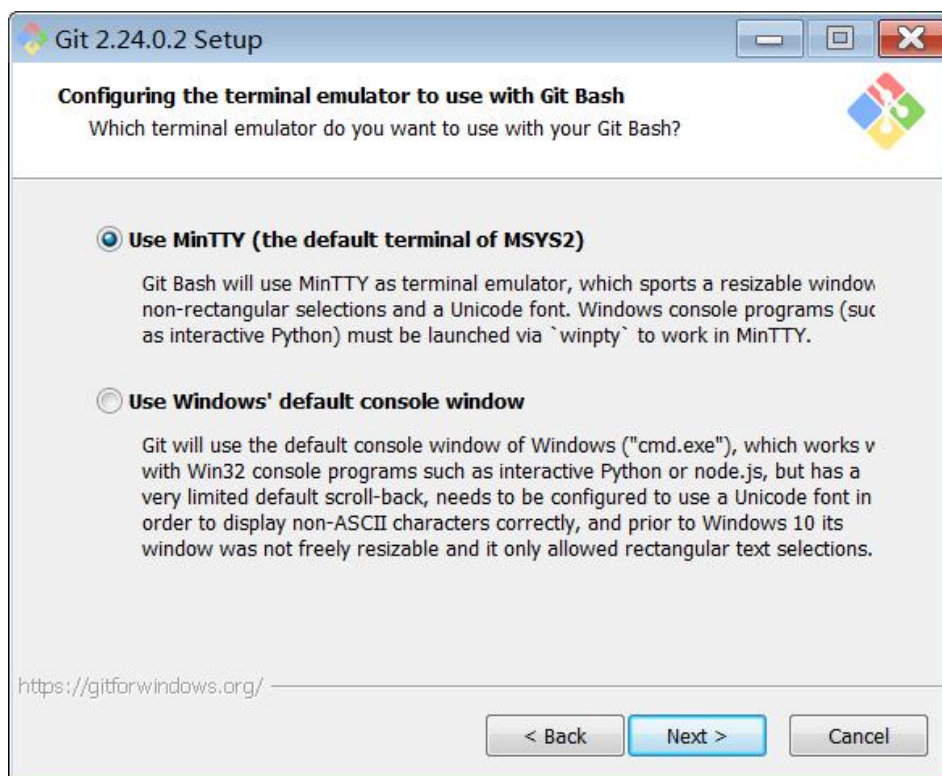


这个界面是设置 PATH 环境我们可以选择配置，选择第二种配置，我们可以从 Git Bash，命令提示符和 Windows PowerShell 以及在 PATH 中寻找 Git 的任何第三方软件中使用 Git，这里推荐选择第二种配置。

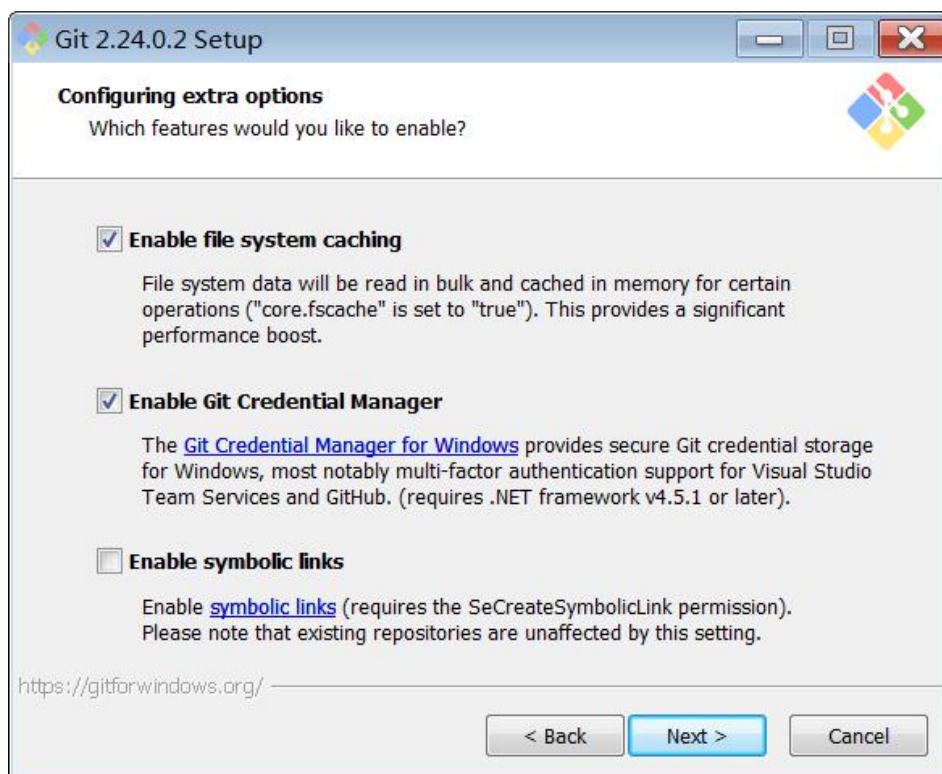
(5) 选择换行格式



(6) 配置 Git Bash 的终端模拟器

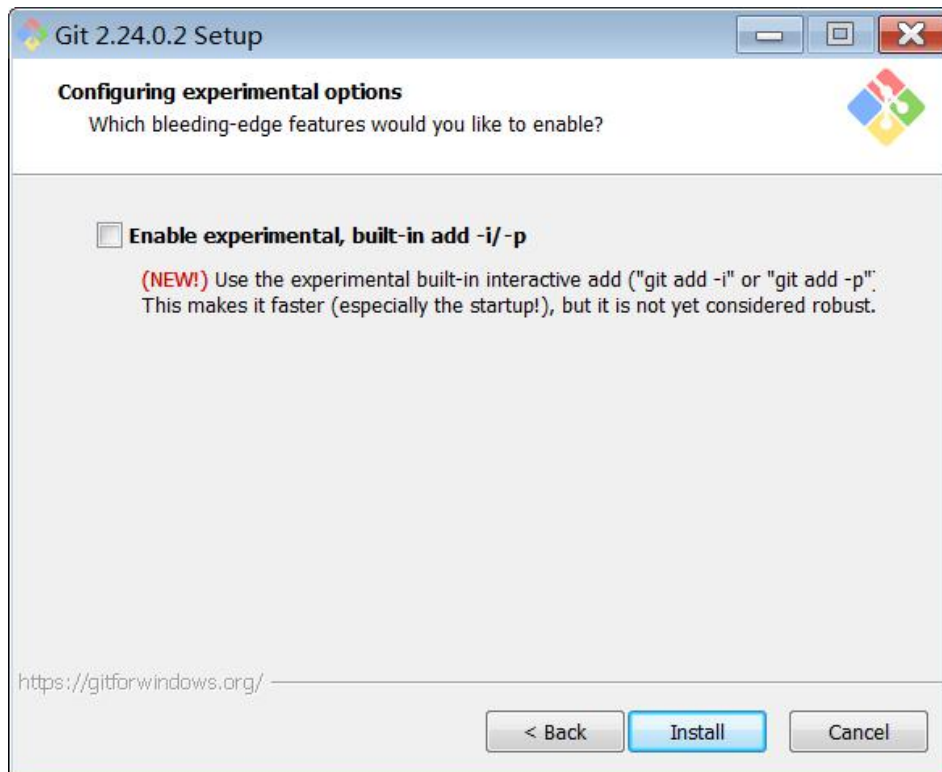


(7) 配置额外的选项

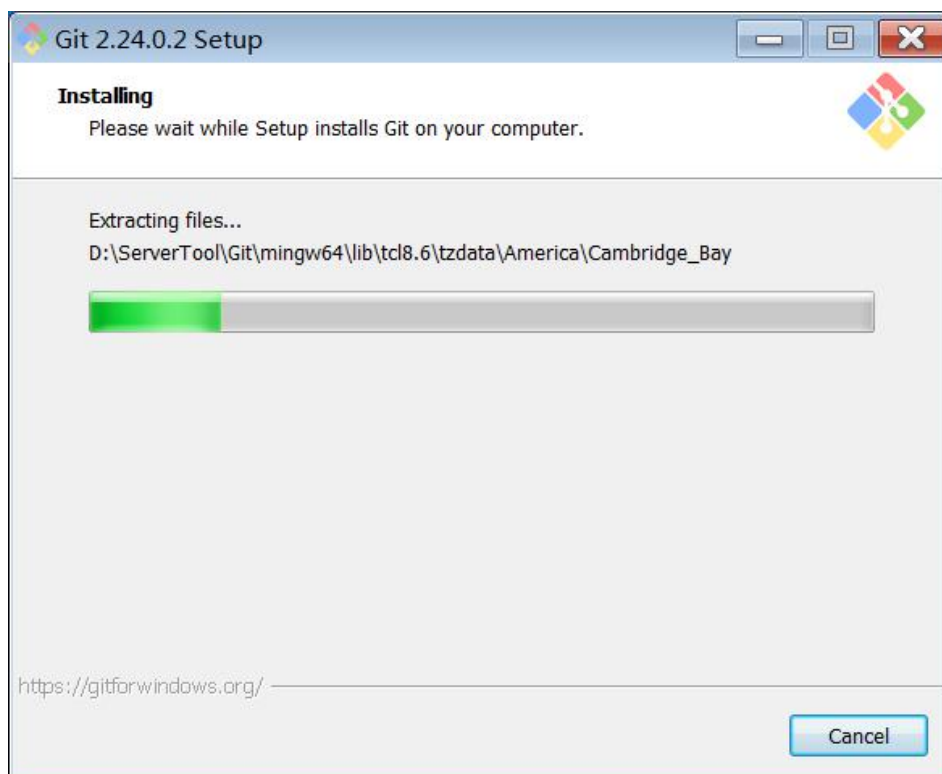


我们可以选择启用文件系统缓存。

(8) 配置实验性选项



(9) 安装进度指示



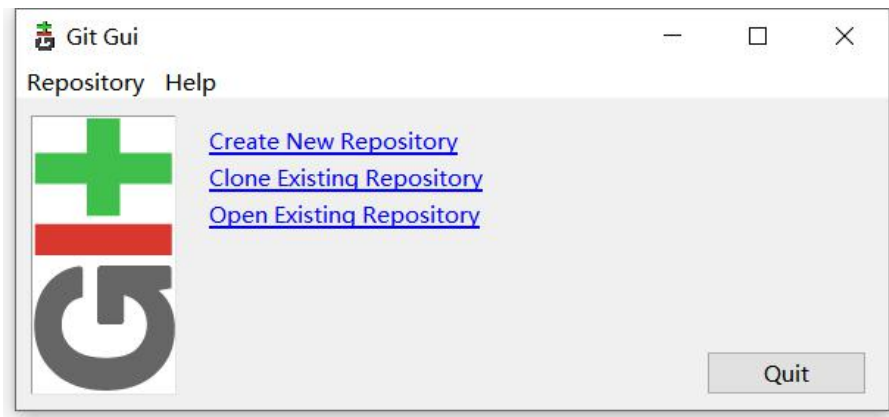
(10) 安装完成



等待安装完后，我们可以在之后的界面选择立即启动 Git Bash 和 查看当前的版本说明。

4.2 创建本地仓库

(1) 首先我们创建后一个目录，然后在文件夹里面右键点击打开 Git Gui，选择 create New Repository。



(2) 选择仓库路径，点击 create 后目录会出现.git 文件。



(3) 也可以右键打开 git bash, 输入 git init 命令也会生成.git 文件

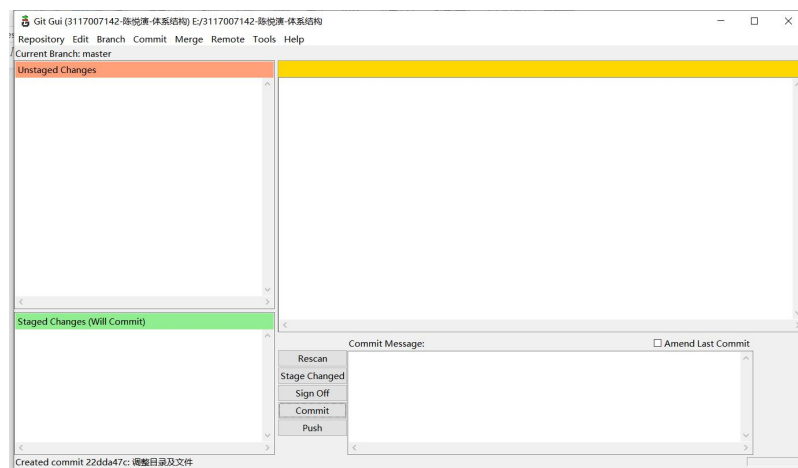


```
MINGW64:/e/3117007142- 陈悦演- 体系结构

YYcoder@DESKTOP-6P68BUG MINGW64 /e/3117007142- 陈悦演- 体系结构
$ git init
Initialized empty Git repository in E:/3117007142- 陈悦演- 体系结构/.git/

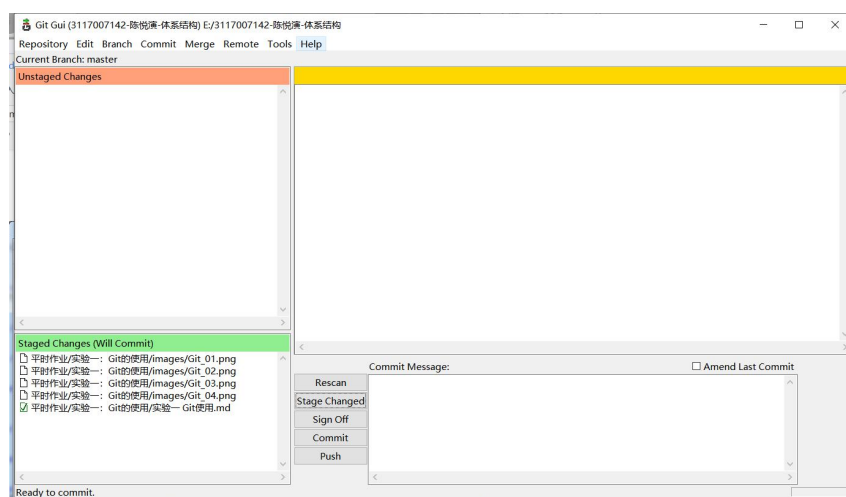
YYcoder@DESKTOP-6P68BUG MINGW64 /e/3117007142- 陈悦演- 体系结构 (master)
$
```

(4) 产生.git 表示本地仓库创建成功, 进入 git gui 界面就可以进行相关可视化的操作了。

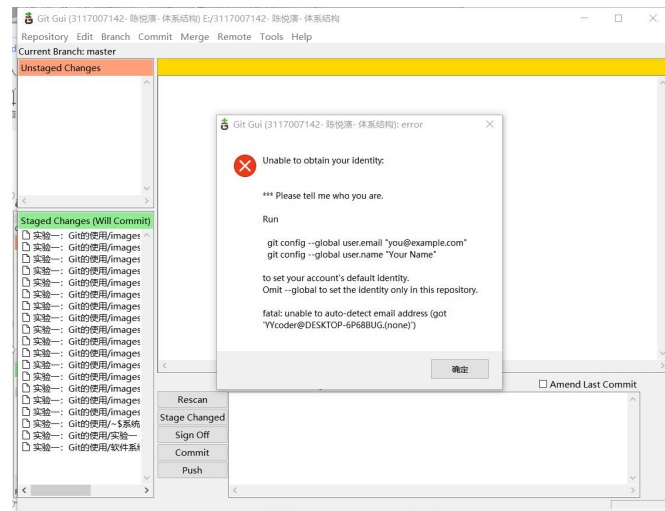


4.3 Git 的提交

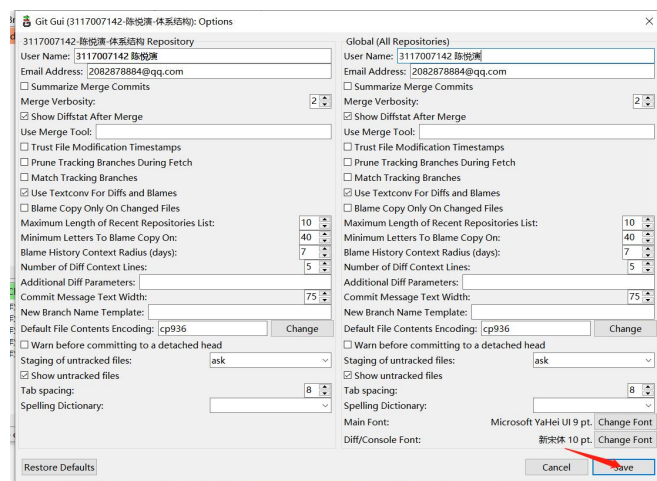
1. 把我们想 commit 的文件加入到 暂存区 Staged Changes(Will commit),可以直接点击 Stage Changed。



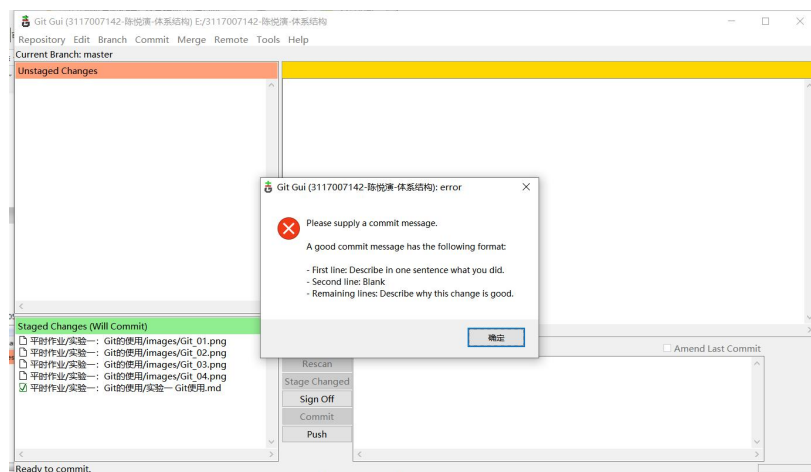
2. 如果我们这时候直接点击 **Commit**, 会发现 **Commit** 失败, 这是因为我们没有输入我们的用户信息。可以用以下方法解决:



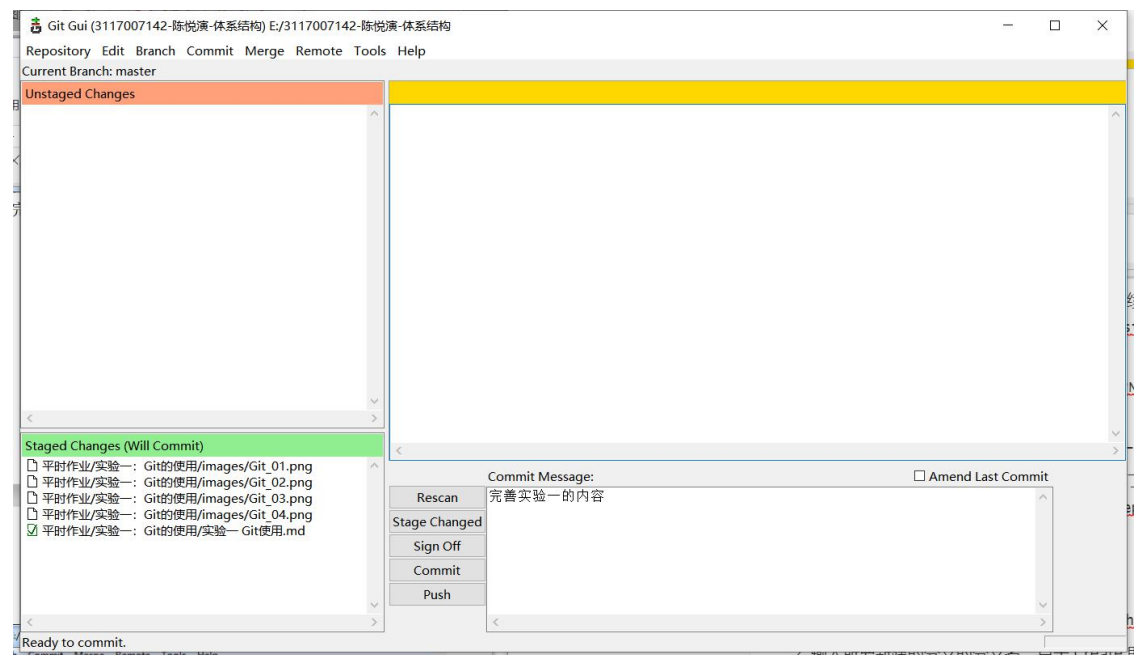
点击菜单栏上的 **Edit**, 选择 **Options**, 然后输入 **User Name** 和 **Email Address**, 最后点击 **save** 之后便可以了。



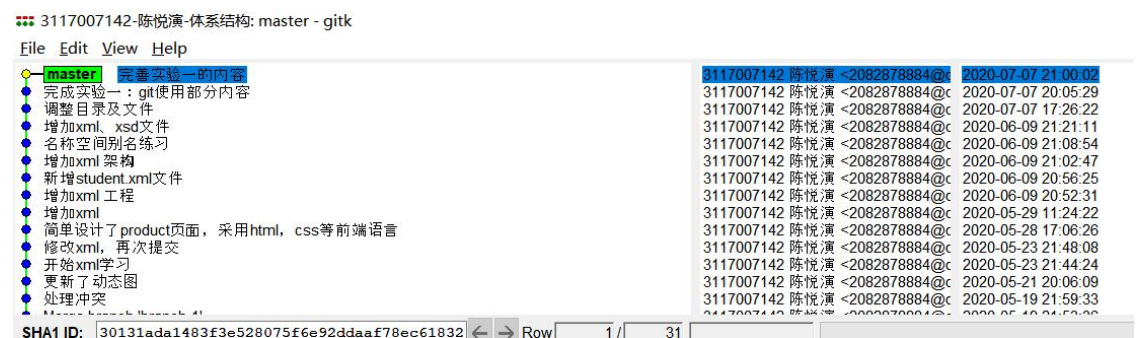
3. 接着, 我们再次点击 **Commit** 之后, 我们会发现还是提交失败, 这是因为我们没有填写 **Commit Message**。



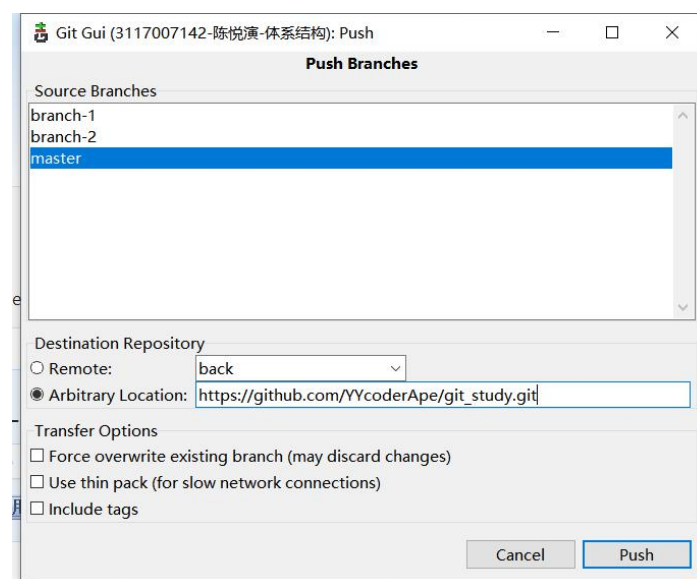
4. 填写完 Commit Message 之后我们便可以继续点击提交。



5. 点击 commit 进行提交，可以在历史记录进行查看提交记录。



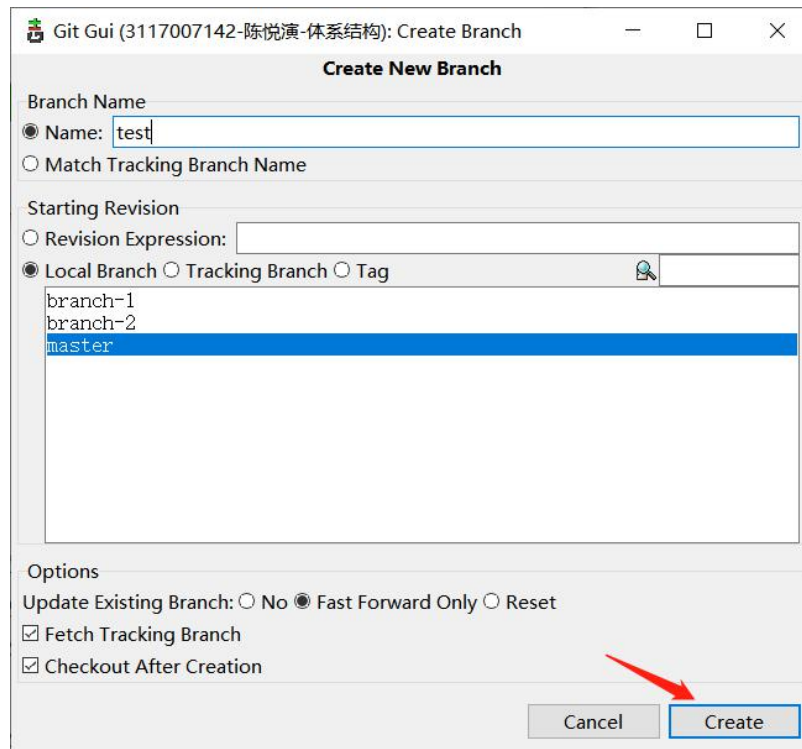
6. Commit 之后我们点击 Push，选择提交的分支。



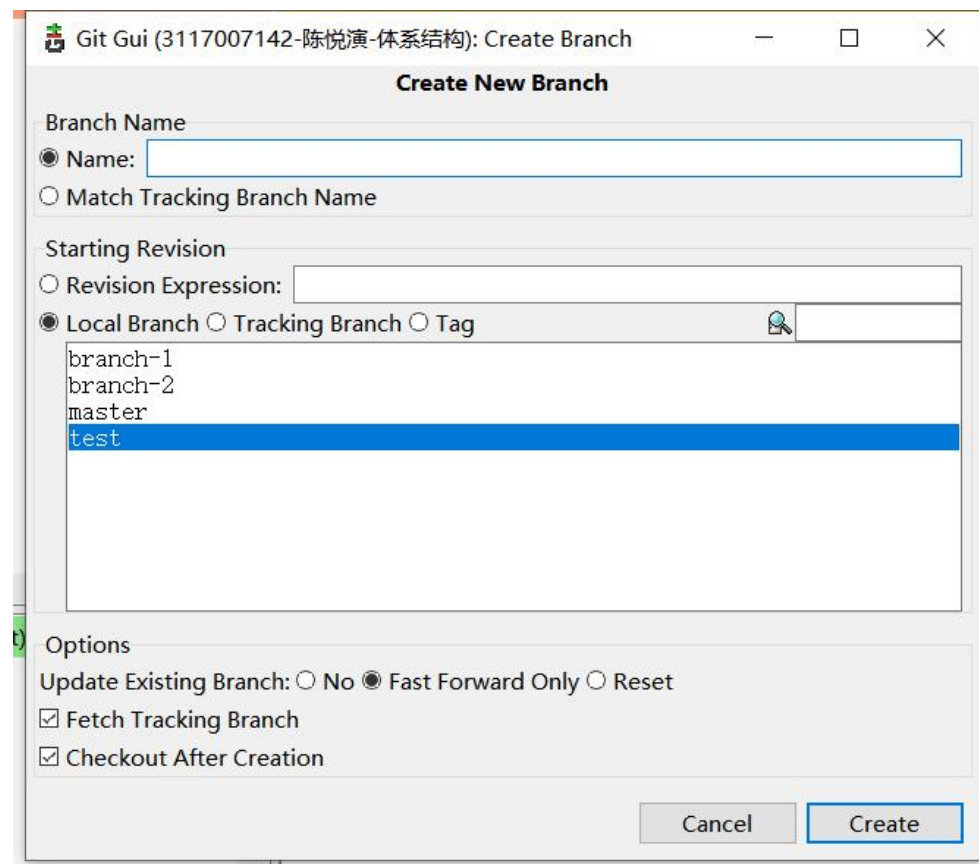
输入用户名和密码之后便成功地 Push 到远端了。

4.3 创建分支

1. 点击菜单栏上的 Branch，输入相应的分支名字，并点击 Create 进行创建

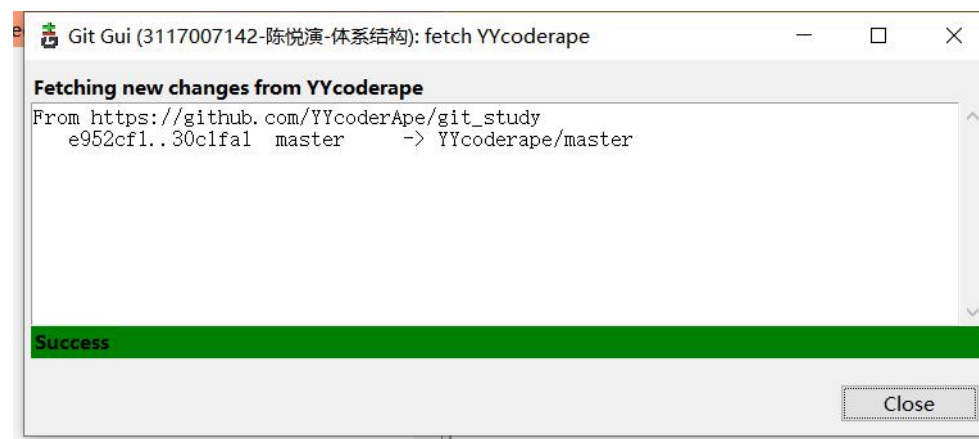
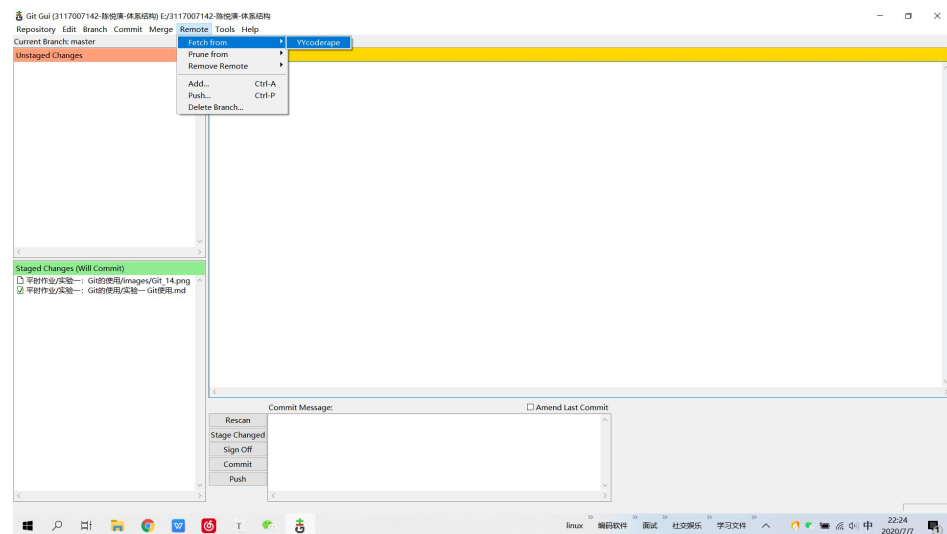


2. 创建完成后，分支列表显示该分支。

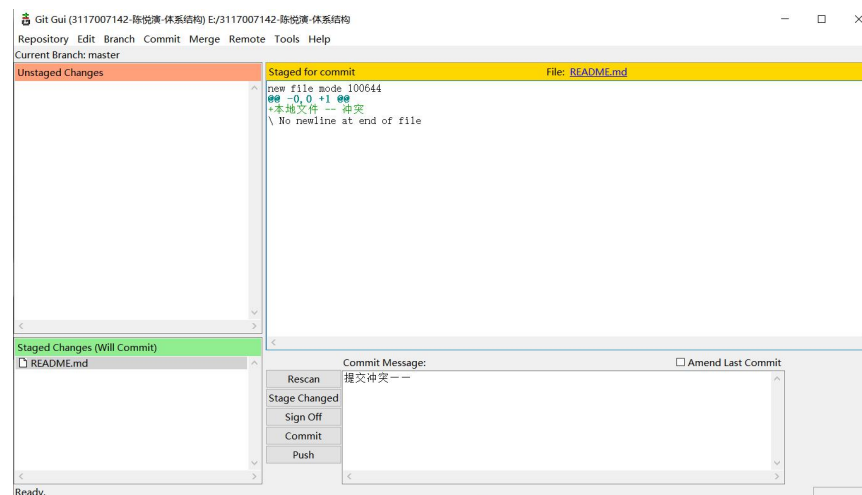


4.4 Git 冲突

1 从远端拉取别人上传的内容,输入用户名和密码之后便拉取成功。

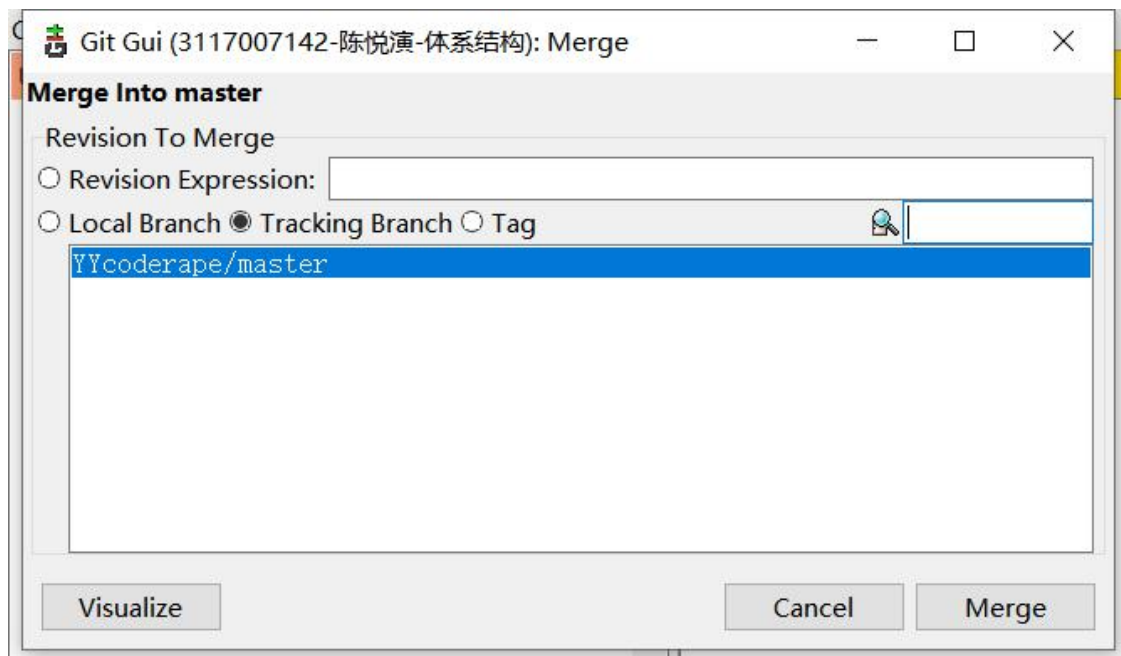


2.进行会产生冲突的操作, master 本地 commit 新的文件, 然后 push 提交到远端, 会发现提交失败。因为远端的版本比我的版本要新(别人有提交新的内容), 我需要先下拉远端最新的版本, 然后点击 Merge 合并。





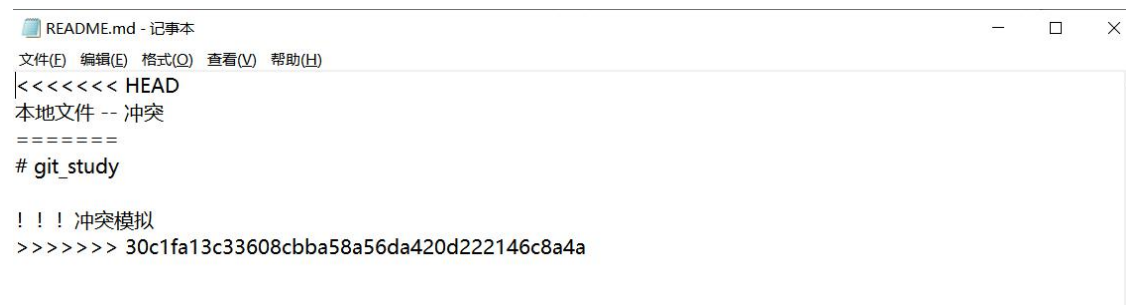
3.在菜单上点击 Merge 进入界面，选择要合并的分支。



4.合并失败。



5.查看冲突信息。



```
README.md - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
<<<<<<< HEAD
本地文件 -- 冲突
=====
# git_study

!!! 冲突模拟
>>>>>> 30c1fa13c33608cbba58a56da420d222146c8a4a
```

6.解决冲突。



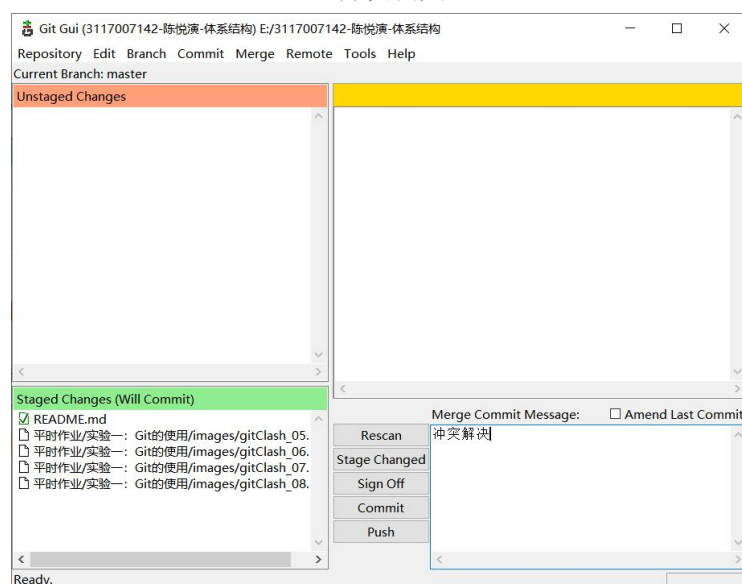
```
README.md - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

本地文件 -- 冲突

# git_study|

!!! 冲突模拟
```

7.再次提交



8.提交成功，意味着冲突已经解决



五、实验总结

通过对 Git 工具的使用，让我体会到了它的强大之处，它可以有效的运用于团队之间的协作开发。经过这次实验中的 Git 操作的练习，让我理解到了分布式版本控制系统确实较于集中式版本控制系统更为好用以及有效率，每个开发者都拥有自己的版本控制库，在自己的版本库上可以任意的执行提交代码、创建分支、拉取合并等行为。在实验中初步掌握了 Git 的安装和简单使用，增加文件、提交、创建分支、增加远程、下拉、上推、合并，能比较熟悉地使用 Git 命令或者 Git GUI 对自己的仓库进行版本管理。

在 Git 的操作过程中遇到的合并冲突问题让我印象非常深刻，因为不同合作者在操作同一个文件的时候，修改之后拉取合并可能会产生冲突，然后我也学会了去查看冲突信息，找到发生冲突的地方，并作出修改解决冲突，最后才能再次提交并推送到远程仓库。通过一系列的操作，最终解决了合并存在的冲突

经过本次的实验，结合课堂上学习到的理论知识，不断地进行实践，让我对 Git 地操作更加熟悉，也加深了对理论知识的理解，整个过程的学习让我收获了不少知识。